(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 307 000 A1

(12) EUROPEAN PATENT APPLICATION

published in accordance with Art. 158(3) EPC

(43) Date of publication:
02.05.2003 Bulletin 2003/18

(21) Application number: **02707171.1**

(22) Date of filing: **27.03.2002**

(51) Int Cl.[7]: **H04L 9/08**, G06F 17/60

(86) International application number:
PCT/JP02/02957

(87) International publication number:
WO 02/080447 (10.10.2002 Gazette 2002/41)

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE TR

(30) Priority: **29.03.2001 JP 2001094808**

(71) Applicant: **Sony Corporation**
Tokyo 141-0001 (JP)

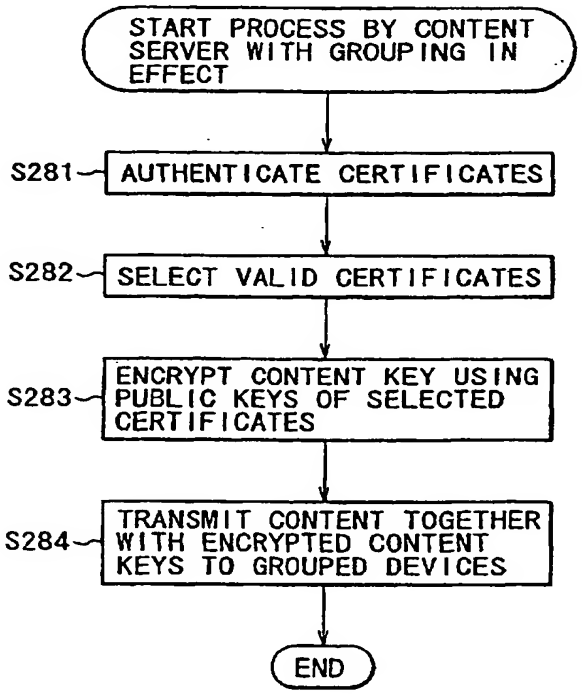
(72) Inventor: **ISHIGURO, Ryuji,**
c/o SONY CORPORATION
Tokyo 141-0001 (JP)

(74) Representative: **Horner, David Richard et al**
D Young & Co,
21 New Fetter Lane
London EC4A 1DA (GB)

(54) INFORMATION PROCESSING APPARATUS

(57) This invention relates to an information processing apparatus for permitting so-called grouping without recourse to group keys. A content server retains in advance certificates of devices subject to grouping. Each certificate contains a public key of the corresponding device. When providing a content, the content server authenticates the certificates of the grouped devices for which the content is destined (step S281), encrypts a content key by use of public keys of the authenticated certificates (step S283), and transmits the content key thus encrypted to each of the devices making up the group (step S284) together with the content. The inventive apparatus is applied to devices that provide contents.

FIG. 39



EP 1 307 000 A1

Fig. 11 is a flowchart of steps constituting a license renewing process performed by the license server in Fig. 1;
Fig. 12 is an explanatory view of a key structure;
Fig. 13 is an explanatory view of a category node arrangement;
Fig. 14 is a schematic view illustrating typical correspondences between nodes and devices;
Fig. 15A is an explanatory view of an enabling key block structure;
Fig. 15B is an explanatory view of another enabling key block structure;
Fig. 16 is an explanatory view showing how an enabling key block is utilized;
Fig. 17 is a schematic view indicating a typical format of the enabling key block;
Fig. 18 is an explanatory view depicting a tag structure of the enabling key block;
Fig. 19 is an explanatory view sketching a content decrypting process using a device node key (DNK);
Fig. 20 is a schematic view of a typical enabling key block;
Fig. 21 is an explanatory view picturing how a plurality of contents are assigned to a single device;
Fig. 22 is an explanatory view of license categories;
Fig. 23 is a timing chart explaining how a registering process is carried out;
Fig. 24 is a flowchart of steps constituting a ripping process performed by the client;
Fig. 25 is an explanatory view of a watermark structure;
Fig. 26 is a schematic view of a typical content format;
Fig. 27 is a schematic view of a typical public key certificate;
Fig. 28 is an explanatory view showing how a content is distributed;
Fig. 29 is a flowchart of steps constituting a content check-out process performed by the client;
Fig. 30 is an explanatory view depicting how enabling key blocks are traced by tag;
Fig. 31 is a schematic view of a typical enabling key block structure;
Fig. 32 is an explanatory view of a mark structure;
Fig. 33 is a flowchart of steps constituting a license purchasing process performed by the client;
Fig. 34 is a flowchart of steps constituting a license purchasing process performed by the license server;
Fig. 35 is a schematic view of a typical mark structure;
Fig. 36 is a flowchart of steps constituting a certificate registering process performed by the client;
Fig. 37 is a flowchart of steps constituting a certificate registering process performed by the content server;
Fig. 38 is a schematic view of typical group certificates;
Fig. 39 is a flowchart of steps constituting a process performed by the content server where grouping is in effect;
Fig. 40 is a schematic view of an encrypted content key;
Fig. 41 is a flowchart of steps constituting a process performed by a client belonging to a group;
Fig. 42 is a flowchart of steps constituting a process performed by a client checking out a license to another client;
Fig. 43 is a flowchart of steps constituting a process performed by a client having a license checked out from another client;
Fig. 44 is a flowchart of steps constituting a reproducing process performed by a client having a license checked out thereto;
Fig. 45 is a flowchart of steps constituting a process performed by a client having a license checked in from another client;
Fig. 46 is a flowchart of steps constituting a process performed by a client having a license checked in to another client;
Fig. 47 is an explanatory view showing how a message authentication code (MAC) is generated;
Fig. 48 is an explanatory view outlining a decrypting process of an integrity check value (ICV) generation key;
Fig. 49 is an explanatory view illustrating another decrypting process of the ICV generation key;
Fig. 50A is an explanatory view depicting how a license copying process is managed with ICV;
Fig. 50B is another explanatory view indicating how the license copying process is managed with ICV; and
Fig. 51 is an explanatory view showing how licenses are managed.

BEST MODE FOR CARRYING OUT THE INVENTION

[0013] Fig. 1 outlines a typical configuration of a content providing system according to the invention. Clients 1-1 and 1-2 (simply called the client 1 hereunder if there is no need for distinction therebetween) are connected to the Internet 2. Although only two clients are shown configured in the example of Fig. 1, any number of clients may be connected to the Internet 2 in practice.

[0014] The Internet 2 is also connected with a content server 3, a license server 4, and an accounting server 5. The content server 3 provides contents to the client 1. The license server 4 offers the client 1 licenses for using the contents provided by the content server 3. The accounting server 5 performs an accounting process regarding the client 1 having acquired a license.

[0015] Any number of content servers 3, license servers 4, and accounting servers 5 may be configured and connected to the Internet 2 in practice.

[0016] Fig. 2 shows a typical structure of the client 1.

[0017] In Fig. 2, a CPU (central processing unit) 21

[0034] The content information includes a content ID (CID) as information for identifying the content data formatted as data, and a codec method for coding and decoding the content in question.

[0035] The URL is information denoting the address to be accessed in acquiring the license designated by the license ID. Illustratively with the system of Fig. 1, the URL stands for the address of the license server 4 from which to acquire licenses. The license ID identifies the license to be needed in utilizing the relevant content recorded as data.

[0036] The data part comprises any number of encryption blocks. Each encryption block is made up of an initial vector (IV), a seed, and data EK'c(data) obtained by encrypting the content data using a key K'c.

[0037] The key K's is constituted by a value obtained by applying the content key Kc and a randomly established seed (value) to a hash function, as defined by the following expression:

K'c = Hash(Kc, Seed)

[0038] Each encryption block is furnished with a different initial vector (IV) and a different seed.

[0039] The encryption of content data is carried out in units of eight bytes. Each eight-byte portion is encrypted by use of the encrypted result from the preceding eight-byte portion in what is known as CBC (cipher block chaining) mode.

[0040] In CBC mode, the first eight-byte content data portion cannot be encrypted using the encrypted result from the preceding eight-byte portion. Instead, the first eight-byte portion is encrypted by use of the initial vector IV as the initial value.

[0041] With CBC mode in effect, even if any one encryption block is unlawfully decrypted, the other encryption blocks will not be decrypted correspondingly.

[0042] This encryption scheme will be described later in more detail by referring to Fig. 47.

[0043] This encryption scheme, it should be noted, is not limitative of the invention. Alternatively, the content data may be encrypted by simply utilizing the content key Kc.

[0044] In the manner described, the client 1 can acquire content data unrestrainedly and free of charge from the content server 3. That is, large quantities of contents can be distributed in a fairly unconstrained manner.

[0045] However, before using any acquired content, each client 1 must be in possession of a license corresponding to the content. How the client reproduces a content will now be described by referring to Fig. 6.

[0046] In step S41, the CPU 21 of the client 1 acquires content ID information (CID) designated by the user operating the input unit 26. The ID information may be constituted illustratively by a content title and a number unique to each of the stored contents.

[0047] When a given content is designated, the CPU 21 reads a license ID relative to the content (i.e., ID of the license for granting the use of the content). The license ID is described in the header of the encrypted content data, as depicted in Fig. 5.

[0048] In step S42, the CPU 21 determines whether or not the license corresponding to the license ID retrieved in step S41 has already been acquired by the client 1 and stored in the storage unit 28. If the license has yet to be acquired, the CPU 21 goes to step S43 and performs a license acquiring process. Details of the license acquiring process will be described later with reference to the flowchart of Fig. 7.

[0049] If in step S42 the license is judged to have been acquired already or if the license acquiring process is carried out in step S43, then step S44 is reached. In step S44, the CPU 21 judges whether or not the acquired license falls within the corresponding expiration date. Whether or not the license has expired is determined by comparing the expiration date stipulated in the license (which will be described later by referring to Fig. 8) with the current date and time kept by the timer 20. If the license is judged to have expired, the CPU 21 goes to step S45 and performs a license renewing process. Details of the license renewing process will be described later by referring to the flowchart of Fig. 10.

[0050] If in step S44 the license is judged to be effective or if the license is renewed in step S45, then step S46 is reached. In step S46, the CPU 21 reads the applicable encrypted content data from the storage unit 28 and places the retrieved data into the RAM 23. In step S47, the CPU 21 supplies the encryption/decryption unit 24 with the content data stored in the RAM 23 in units of encryption blocks as shown in Fig. 5, and causes the unit 24 to decrypt the data using the content key Kc.

[0051] The content key Kc is obtained (to be described later in more detail by referring to Figs. 15A and 15B) illustratively as follows: a key K_{EKBC} is first acquired using a device node key (DNK). The content key Kc is then obtained from the data K_{EKBC}(Kc) (see Fig. 5) by use of the acquired key K_{EKBC}.

[0052] In step S48, the CPU 21 supplies the codec unit 25 with the content data decrypted by the encryption/decryption unit 24, and causes the codec unit 25 to decode the supplied data. The CPU 21 then sends the data decoded by the codec unit 25 to the output unit 27 through the I/O interface 32. In turn, the output unit 27 converts the received digital data to analog format for audio output through the speakers.

[0053] How the license acquiring process is performed in step S43 of Fig. 6 will now be described in detail with reference to the flowchart of Fig. 7.

[0054] The client 1 accesses the license server 4 in advance for a registering process whereby service data are acquired, including a leaf ID, a DNK (device node key), a private key paired with a public key for the client 1, a public key of the license server 4, and certificates of the respective public keys. The registering process by the client 1 will be described later in detail by referring to Fig. 23.

[0055] The leaf ID represents identification informa-

step S107, the CPU 21 selects the use conditions corresponding to the license selected in step S105. If the use conditions were designated in step S102 by the user, the designated use conditions may be added as needed to the previously prepared use conditions. The CPU 21 furnishes the license with the use conditions thus selected.

[0069] In step S108, the CPU 21 affixes a digital signature to the license by use of a private key from the license server 4. This step generates a license whose structure is shown in Fig. 8.

[0070] In step S109, the CPU 21 of the license server 4 transmits the license (shown structurally in Fig. 8) to the client 1 through the communication unit 29 and over the Internet 2.

[0071] In step S110, the CPU 21 of the license server 4 places into the storage unit 28 the license that has just been transmitted (including the use conditions and leaf ID) in correspondence with the user ID and password acquired in step S102. In step S111, the CPU 21 carries out an accounting process. More specifically, through the communication unit 29, the CPU 21 requests the accounting server 5 to carry out an accounting process regarding the user corresponding to the user ID and password. Given the accounting request, the accounting server 5 bills the user for the license. If the user fails to pay the billed amount, the user from then on will be banned from acquiring any further license that may be requested.

[0072] In such a case, the accounting server 5 returns in step S104 the credit rejection data banning the granting of the requested license. Step S104 is then followed by step S112 in which the CPU 21 performs error handling. More specifically, the CPU 21 of the license server 4 outputs to the client 1 having gained access through the communication unit 29 a message saying that the license cannot be granted to the user. The CPU 21 then terminates the process.

[0073] In this case, the user cannot utilize the content (i.e., unable to decrypt the content), having failed to receive the license for the reason above.

[0074] Fig. 10 is a flowchart of detailed steps constituting a license renewing process in step S45 of Fig. 6. Steps S131 through S135 in Fig. 10 are basically the same as steps S61 through S65 in Fig. 7, except that in step S133 the CPU 21 acquires the ID of the license that is not to be purchased but to be renewed. In step S135, the CPU 21 transmits to the license server 4 the ID of the license to be renewed together with the user ID and password.

[0075] In response to the transmission from the client 1 in step S135, the license server 4 proposes use conditions as will be described later (in step S153 of Fig. 11). In step S136, the CPU 21 of the client 1 receives the proposed use conditions from the license server 4 and forwards the received conditions to the output unit 27 for display. The user may select desired use conditions from those proposed or may add new conditions thereto by operating the input unit 26. In step S137, the CPU 21 transmits to the license server 4 sign-up data for purchasing the selected use conditions (i.e., conditions for renewing the license). Upon receipt of the sign-up data, the license server 4 returns definitively proposed use conditions (in step S154 of Fig. 11). In step S138, the CPU 21 of the client 1 acquires the use conditions from the license server 4. In step S139, the CPU 21 substitutes the newly acquired use conditions for the currently stored use conditions corresponding to the license in the storage unit 28.

[0076] Fig. 11 is a flowchart of steps constituting a license renewing process performed by the license server 4 in conjunction with the license renewing process carried out by the client 1 as described above.

[0077] In step S151, the CPU 21 of the license server 4 is first accessed by the client 1. In step S152, the CPU 21 receives the license-designating information transmitted by the client 1 in step S135 together with a license renewal request.

[0078] In step S153, given the license renewal request, the CPU 21 retrieves from the storage unit 28 the use conditions (to be renewed) corresponding to the license in question. The retrieved use conditions are transmitted to the client 1.

[0079] Upon receipt of the use conditions thus proposed, the client 1 signs up for the purchase of the conditions in step S137 of Fig. 10 as described above. In step S154, the CPU 21 of the license server 4 generates data corresponding to the use conditions that the client 1 has signed up to purchase, and transmits the generated data to the client 1. In turn, the client 1 renews the use conditions of the currently registered license by utilizing the use conditions received in step S139 as described above.

[0080] The inventive system, as shown in Fig. 12, manages the keys of devices and licenses based on the principle of what is known as broadcast encryption (refer to Japanese Patent Laid-open No. 2001-352321). The keys make up a hierarchical tree structure in which the leaves at the bottom level correspond to the keys of individual devices. In the example of Fig. 12, keys are generated to represent 16 devices or licenses numbered 0 through 15.

[0081] Each key is defined so as to correspond with each of the nodes (shown as circles in the figure) constituting the tree structure. In this example, a root key KR denotes the root node at the highest level; keys K0 and K1 correspond to the nodes at the second-highest level; keys K00 through K11 represent the nodes at the third-highest level; and keys K000 through K111 match the nodes at the fourth-highest level. Keys K0000 through K1111 correspond to the leaves representative of the nodes at the bottom level (i.e., device nodes).

[0082] In this hierarchical structure, the key immediately above, say, keys K0010 and K0011 is a key K001; and the key immediately above keys K000 and K001 is a key K00. In like manner, keys K00 and K01 are topped

sumed in the highest node, a manufacturer, a content provider or any other organization managing the tree structure made up of these nodes may generate uniquely defined enabling key blocks (EKB) each covering nodes leading up to the highest-level node and may distribute the generated blocks to any devices belonging to the subordinate nodes under the topmost node. In that setup, any key may be renewed in a manner totally independent of the devices belonging to any other category except the topmost node.

[0093] For example, in the tree structure of Fig. 12, four devices 0, 1, 2 and 3 contained in a group possess common keys K00, K0 and KR as their node keys. This shared node key structure may be utilized in providing a common content key to the devices 0, 1, 2 and 3 only. If the shared node key K00 is established as a content key, only the devices 0, 1, 2 and 3 may be assigned the common content key without being furnished with any new key. As another example, suppose that a new content key Kcon is encrypted using the node key K00 to generate a value Enc(K00, Kcon) which is then distributed to the devices 0, 1, 2 and 3 over a network or by use of suitable storage media. In that case, solely the devices 0, 1, 2 and 3 can acquire the content key Kcon by decrypting the encrypted value Enc(K00, Kcon) using the shared node key K00. The notation Enc(Ka, Kb) represent the data obtained by encrypting data Kb with data Ka.

[0094] Suppose that at a given point "t" the keys K0011, K001, K00, K0 and KR owned by the device 3 are found to be exposed by a hacker through analysis. In that case, the device 3 needs to be isolated from the system in order to protect data exchanged within the system (i.e., in the group of devices 0, 1, 2 and 3). This requires replacing the node keys K001, K00, K0 and KR with new keys K(t)001, K(t)00, K(t)0 and K(t)R respectively and informing the devices 0, 1 and 2 of the renewed keys. The notation K(t)aaa indicates a renewed key of a generation "t" derived from a key Kaaa.

[0095] How renewed keys are distributed will now be described. The key renewal process is carried out illustratively by furnishing the devices 0, 1 and 2 with a table composed of block data called an enabling key block (EKB), shown in Fig. 15A, distributed over the network or by use of suitable storage media. Each enabling key block (EKB) is constituted by encryption keys that are used to distribute renewed keys to the devices corresponding to the leaves (i.e., nodes at the bottom level) in the tree structure such as that in Fig. 12. The enabling key block (EKB) may also be called a key renewal block (KRB).

[0096] The enabling key block (EKB) shown in Fig. 15A constitutes block data having a data structure in which only the devices needing to have their node keys renewed are allowed to do so. The example of Fig. 15A is the block data prepared in such a manner as to distribute renewed node keys of the generation "t" to the devices 0, 1 and 2 in the tree structure of Fig. 12. As is evident in Fig. 12, the devices 0 and 1 need renewed node keys K(t)00, K(t)0 and K(t)R while the device 2 requires renewed node keys K(t)001, K(t)00, K(t)0 and K(t)R.

[0097] As indicated by the EKB in Fig. 15A, each EKB contains a plurality of encryption keys. The encryption key in the bottom row of Fig. 15A is Enc(K0010, K(t)001) representative of the renewed node key K(t)001 encrypted by use of the leaf key K0010 owned by the device 2. The device 2 acquires the renewed node key K(t)001 by decrypting the encryption key Enc(K0010, K(t)001) using its own leaf key K0010. The renewed node key K(t)001 obtained through such decryption may then be used to decrypt another encryption key Enc(K(t)001, K(t)00) in the second row from the bottom of Fig. 15A; the decryption yields another renewed key K(t)00.

[0098] Likewise, another encryption key Enc(K(t)00, K(t)0) in the second row from the top in Fig. 15A is decrypted to provide a renewed node key K(t)0; the renewed node key K(t)0 is then used to decrypt another encryption key Enc(K(t)0, K(t)R) in the topmost row of Fig. 15A to produce a renewed root key K(t)R.

[0099] Meanwhile, the node key K000 is not subject to renewal. What the nodes 0 and 1 need as renewed node keys are the keys K(t)00, K(t)0 and K(t)R. The nodes 0 and 1 acquire the renewed node key K(t)00 by decrypting the encryption key Enc(K000, K(t)00) in the third row from the top in Fig. 15A using the node key K000 included in the device node keys. In like manner, the encryption key Enc(K(t)00, K(t)0) in the second row from the top in Fig. 15A is decrypted so as to provide the renewed node key K(t)0; the encryption key Enc(K(t)0, K(t)R) in the top row of Fig. 15A is decrypted in order to produce the renewed root key K(t)R. This is how the devices 0, 1 and 2 can obtain the renewed key K(t)R.

[0100] In Fig. 15, each of the indexes in the left-hand side column stands for an absolute address of a node key or a leaf key used as the decryption key for decrypting the corresponding encryption key listed in the right-hand side column.

[0101] Suppose that renewal of the node keys K(t)0 and K(t)R at the upper levels of the tree structure in Fig. 12 is not necessary and that only the node key K00 needs to be renewed. In that case, the enabling key block (EKB) of Fig. 15B may be used to distribute the renewed node key K(t)00 to the devices 0, 1 and 2.

[0102] The EKB shown in Fig. 15B is effective where a renewed content key is distributed so as to be shared within a specific group of devices. As an example, suppose that the devices 0, 1, 2 and 3 forming a group enclosed by dotted lines in Fig. 12 utilize a particular storage medium and that they need a renewed common content key K(t)con. In that case, a renewed node key K(t)00 is first derived from the node key K00 shared by the devices 0, 1, 2 and 3. The renewed node key K(t)00 is then used to encrypt the renewed content key K(t)con, generating data Enc(K(t)00, K(t)con). The encrypted data Enc(K(t)00, K(t)con) are distributed to the relevant de-

responding to the key K_{EKBC} in Fig. 5). The encrypted content key together with the EKB is added to the encrypted content that is offered to the client 1.

[0114] The EKB in Fig. 19 contains the root key KR that can be decrypted using a DNK (device node key) as shown in Fig. 20. The client 1 can thus obtain the root key KR from the EKB using the DNK contained in the service data. The content key Kc is decrypted from the encrypted content key Enc(KR, Kc) by use of the root key KR. The content key Kc is then used to decrypt the content from the encrypted data Enc(Kc, Content).

[0115] As described, where the DNK is assigned individually to each client 1, the license of a given client 1 can be revoked independently on the basis of the principle described above with reference to Figs. 12, 15A and 15B.

[0116] When each license is distributed together with a leaf ID to the client 1, the client 1 brings the service data into correspondence with the license. Such correspondence helps prevent illegal copying of the license.

[0117] Where a client-addressed certificate and a private key are distributed as the service data to each client, the end user at the client can prepare a content that is resistant to illegal copying through the use of the service data.

[0118] How the certificate and private key are utilized will be described later with reference to the flowchart of Fig. 29.

[0119] As described above with reference to Fig. 13, a given category node may be established to bring the inventive content distribution system for managing licenses into correspondence with a category of devices that utilize diverse contents. That means the same device may be assigned a plurality of DNKs. It follows that contents of different categories can be managed by a single device.

[0120] Fig. 21 shows how such content management is carried out. In the setup of Fig. 21, a device D1 retains service data and a license for using a content 1 that is assigned DNK1 according to the inventive principle of the content distribution system. At the same time, the device D1 may also have a content 2 placed in a Memory Stick after it is ripped from a CD, the content 2 being assigned DNK2. In this case, the device D1 can concurrently deal with different contents, that is content 1 and content 2, distributed according to different systems (i. e., the content distribution system and the device management system). The arrangement above is not adopted if the currently assigned DNK is deleted before a new DNK is assigned so that the device in question is always associated with a single DNK.

[0121] In the tree structure of Fig. 13, each of the triangles at the lowest 32 levels may illustratively be assigned a license category 1 and a license category 2 shown in Fig. 22. This means that any single category may be divided into subcategories for better management covering detailed data items such as the genre of the content, the disc label involved, the distributor's name, the type of distribution service, the source of the content, and a specific manner in which the content is offered.

[0122] In the example of Fig. 22, a license category 1 is shown covering the genre of jazz and a license category 2 the genre of rock and roll. The license category 1 is matched with contents 1 and 2 which have a license ID of 1 each and which are distributed to users 1, 2 and 3. The license category 2 comprises contents 3, 4 and 5 which having a license ID of 2 each and which are provided to the users 1 and 3.

[0123] In this setup, keys can be managed independently in units of categories according to the invention.

[0124] Also according to the invention, it is possible to have DNKs downloaded through the license server 4 to individual devices or storage media at the time of a registering process, instead of having the DNKs incorporated in devices or embedded on storage media beforehand. This makes it possible to implement a system for allowing users to acquire keys.

[0125] How the above-mentioned registering process is performed by the client 1 will now be described with reference to Fig. 23.

[0126] In step S161, the CPU 21 of the client 1 causes the communication unit 29 to transmit a service data request to the license server 4. In step S165, the CPU 21 of the license server 4 receives the service data request through the communication unit 29. In step S166, the CPU 21 of the license server 4 transmits a user information request to the client 1 through the communication unit 29.

[0127] In step S162, the CPU 21 of the client 1 receives the user information request through the communication unit 29. In turn, the CPU 21 causes the output unit 27 to display a message prompting the user to enter user information. Upon viewing the message, the user operates the keyboard or the like to enter the user information such as the user's personal information and accounting information into the input unit 26. In step S163, the CPU 21 of the client 1 transmits the user-input information to the license server 4 through the communication unit 29.

[0128] In step S167, the CPU 21 of the license server 4 receives the user information through the communication unit 29. In step S168, the CPU 21 assigns the client 1 to any one of the unassigned leaves below the node of the category corresponding to the license server 4, and generates a device node key in the form of a set of node keys assigned to the nodes along the path ranging from the leaf assigned to the client 1 to the node corresponding to the category of the license server 4. The CPU 21 then generates service data by putting together the device node key generated as described, the leaf ID of the leaf assigned to the client 1, a private key of the client 1, a public key paired with the private key of the client 1, a public key of the license server, and certificates of the public keys. In step S169, the CPU 21 of the license server 4 transmits the generated service

cense server 4 acting as the certificate authority. The digital signature is made of data generated by use of the private key of the license server 4 on the basis of a hash value generated by applying a hash function to the message.

[0146] In the example of Fig. 12, the device 0 is assigned a node ID or a lead ID of "0000"; the device 1, the ID of "0001"; and the device 15, the ID of "1111." Such IDs determine where each device (i.e., entity) is positioned (as a leaf or a node) in the tree structure.

[0147] Where the license for granting the use of each content is distributed independent of the content in question, the content can be distributed unrestrainedly. All contents acquired in any manner or through any channels may then be handled in unified fashion.

[0148] If the file format is constituted as shown in Fig. 26, the copyright of each content in that format can be properly controlled not only when the content is distributed over the Internet but also when the content is offered to SDMI (Secure Digital Music Initiative) apparatuses.

[0149] Furthermore, if the content is distributed on a storage medium or over the Internet 2 as shown in Fig. 28. the content can be checked out to a portable device (PD) as an SDMI apparatus by resorting to the process explained above.

[0150] Described below with reference to the flowchart of Fig. 29 is how the client 1 checks out a content to another client (e.g., PD).

[0151] In step S191, the CPU 21 judges whether or not a digital signature is affixed to the content. If the digital signal is judged affixed, step S192 is reached. In step S192. the CPU 21 extracts a certificate from the content and authenticates it using a public key of the certificate authority (i.e., license server 4). More specifically, the client 1 acquires from the license server 4 a public key paired with the private key of the license server 4 and decrypts the digital signature affixed to the public key certificate by use of the acquired public key. As described above with reference to Fig. 27, the digital signature is prepared based on the private key of the certificate authority (license server 4) and thus can be decrypted using the public key of the license server 4. The CPU 21 further computes a hash value by applying a hash function to the whole message in the certificate. The CPU 21 compares the computed hash value with a hash value obtained by decrypting the digital signature. If the two values match, the message is judged to be free of tampering. If the two hash values differ upon comparison, the certificate is judged to have been tampered with.

[0152] In step S193, the CPU 21 checks to see whether or not the certificate has been tampered with. If the certificate is judged to be free of tampering, step S194 is reached in which the certificate is authenticated using the EKB. The authenticating process is carried out by determining whether or not it is possible to effect trace through the EKB based on the leaf ID included in the certificate (Fig. 27). How the authenticating process is performed will now be described with reference to Figs. 30 and 31.

[0153] Suppose now that a device having a leaf key K1001 is a revoked device as shown in Fig. 30. In that case, an EKB having data (encryption keys) and tags shown in Fig. 31 is distributed to each device (leaf). The EKB is arranged so as to renew keys KR, K1, K10 and K100 for revoking the device 1001 in Fig. 30.

[0154] All leaves except the revoked device 1001 can acquire a renewed root key K(t)R. That is, since the leaves below a node key K0 each retain the unrenewed node key K0 within the device, each of these leaves can obtain a renewed root key K(t)R by decrypting an,encryption key Enc(K0, K(t)R) using the key K0.

[0155] The leaves below a node 11 may each acquire a renewed node key K(t)1 by decrypting an encryption key Enc(K11, K(t)1) using a node key K11 yet to be renewed. Furthermore, an updated root key K(t)R may be obtained by decrypting an encryption key Enc(K(t)1, K(t)R) using the node key K(t)1. The leaves below a node key K101 may likewise obtain the renewed root key K(t)R.

[0156] A device 1000 having an unrevoked leaf key K1000 may acquire a node key K(t)100 by decrypting an encryption key Enc(K1000, K(t)100) using its own leaf key K1000. The node key K(t)100 thus acquired is then used successively to decrypt node keys at higher levels until the renewed root key K(t)R is obtained.

[0157] On the other hand, the revoked device 1001 is incapable of acquiring the renewed node key K(t)100 one level higher through the EKB process. That means the renewed root key K(t)R cannot be obtained.

[0158] The valid (i.e., unrevoked) device (client 1) is furnished with the EKB containing the data and tags shown in Fig. 31. The EKB is distributed by the license server 4 to each device for storage therein.

[0159] Each client may carry out an EKB tracing process using the furnished tags. The process involves determining whether or not the key distribution tree may be traced starting from the topmost root key.

[0160] Illustratively, the leaf ID "1001" of the leaf 1001 in Fig. 30 may be regarded as four-bit data (1, 0, 0, 1). A check is then made to see. if the tree structure can be traced starting from the most significant bit. A "1" bit is interpreted to indicate a rightward advance and a "0" bit a leftward advance.

[0161] Because the most significant bit of the ID "1001" is "1," the trace advances right from the root key KR in Fig. 30. The first tag (numbered 0) in the EKB is defined as 0: {0, 0}, interpreted to indicate the presence of data on both branches. Since the rightward advance is in effect in this case, the node key K1 is reached.

[0162] The trace now goes to a node below the node key K1. The second bit in the ID "1001" is 0, indicating a leftward advance. The tag numbered 1 denotes the presence or absence of data below the node key K0 to the left, and the tag numbered 2 represents the pres-

[0180] Described below with reference to the flowchart of Fig. 33 is how a mark is added to a content when the user purchases the license for that content.

[0181] In step S221, the CPU 21 first gains access to the license server 4 over the Internet 2 in response to a command entered by the user into the input unit 26.

[0182] In step S222, the CPU 21 acquires the command input from the user through the input unit 26. In accordance with the command, the CPU 21 requests an outright purchase of the license from the license server 4.

[0183] Upon receipt of the request, the license server 4 proposes a price for the license, as will be described later with reference to the flowchart of Fig. 34 (in step S242 of Fig. 34). In step S223, the CPU 21 of the client 1 receives the proposed price from the license server 4 and causes the output unit 27 to display the price.

[0184] Upon viewing the display, the user decides whether or not to accept the proposed price. The user enters the result of his or her decision into the input unit 26.

[0185] In step S224, the CPU 21 receives the user's input through the input unit 26 and judges whether or not the user has accepted the proposed price. If the proposed price is judged accepted, the CPU 21 goes to step S225 and reports the acceptance to the license server 4.

[0186] Given the report of the acceptance, the license server 4 returns a mark that has an ownership flag, i.e., information denoting the outright purchase of the license at the proposed price, described therein (in step S244 of Fig. 34). In step S226, the CPU 21 of the client 1 receives the mark from the license server 4. In step S227, the CPU 21 embeds the received mark into the content. This causes the mark including the ownership flag of Fig. 32 to be recorded as a mark of content relative to the purchased license in correspondence with the content. With the message thus renewed, the CPU 21 also renews the digital signature (Fig. 26) and writes the renewed signature to the storage medium.

[0187] If in step S224 the price proposed by the license server 4 is not judged accepted, step S228 is reached. In step S228, the CPU 21 reports rejection of the proposed price to the license server 4.

[0188] In conjunction with the above-described process of the client 1, the license server 4 carries out the steps in the flowchart of Fig. 34.

[0189] In step S241, the CPU 21 of the license server 4 first receives a license purchase request from the client 1 (in step S222 of Fig. 33). Upon receipt of the request, the CPU 21 goes to step S242 to retrieve from the storage unit 28 the price for the outright purchase of the license in question, and transmits the price to the client 1.

[0190] As described above, the client 1 reports either the acceptance or the rejection of the proposed price.

[0191] In step S243, the CPU 21 of the license server 4 judges whether or not the report of the acceptance is received from the client 1. If the acceptance report is judged received, then step S244 is reached. In step S244, the CPU 21 of the license server 4 generates a mark that contains a message specifying the purchase of the license in question, affixes a digital signature to the mark using its own private key, and transmits the mark to the client 1. The mark thus transmitted is written to the applicable content in the storage unit 28 of the client 1 as described above (in step S227 of Fig. 33).

[0192] If in step S243 the acceptance report is not judged received from the client 1, then step S244 is skipped. In this case, the purchase of the license is not accomplished, so that the mark will not be transmitted.

[0193] Fig. 35 shows a typical structure of a mark transmitted from the license server 4 to the client 1. In this example, the mark is made up of the user's leaf ID and his or her ownership flag (Own) and of a digital signature Sigs(LeafID, Own) generated using a private key S of the license server 4 on the basis of the leaf ID and ownership flag.

[0194] The mark is valid only for a specific content of a particular user. If the content in question is copied, the mark accompanying the copied content is invalidated.

[0195] As described, each content and its license are handled independently of one another, and the use conditions are associated with each license. This scheme makes it possible to offer diverse services reflecting the different use status of individual contents.

[0196] Described below is what is known as grouping. Grouping involves putting together a plurality of devices or storage media to form a group within which a content may be exchanged freely. Grouping usually applies to devices or storage media owned by an individual. Whereas the devices or storage media forming a single group were conventionally assigned a group key for control purposes, the target devices or storage media to be grouped may be associated with a single license for easier grouping control according to the invention.

[0197] It is also possible to register beforehand each of the devices forming a given group for the same control purpose. Typical grouping with devices registered in advance will now be described.

[0198] In this example, the user needs to register beforehand with the server the certificates of the devices to be grouped. The certificates are registered in the steps of the flowcharts in Figs. 36 and 37.

[0199] Referring first to Fig. 36, the client (one of the devices to be grouped) has its certificate registered as follows: in step S261, the CPU 21 of the client 1 which is subjected to grouping prepares its own certificate containing its public key.

[0200] In step S262, the CPU 21 gains access to the content server 3 based on the user's input through the input unit 26. In step S263. the certificate prepared in step S261 is transmitted to the content server 3.

[0201] Alternatively, the certificate received from the license server 4 may be used unmodified for the registration.

[0202] The steps above are carried out by all devices

[0225] If the check-out count N1 is judged smaller than the maximum check-out count N2, step S304 is reached. In step S304, the CPU 21 acquires the leaf key of the other client (i.e., client of the check-out destination) and writes the acquired leaf key to a check-out list in the storage unit 28 in correspondence with the ID of the license to be checked out.

[0226] In step S305, the CPU 21 increments by 1 the check-out count N1 of the license, the count having been retrieved in step S301. In step S306, the CPU 21 computes an ICV based on the message of the license. The ICV will be described later with reference to Figs. 47 through 51. The ICV scheme is designed to prevent the tampering of the licenses.

[0227] In step S307, the CPU 21 encrypts the license in question as well as the ICV computed in step S306 using the public key of this client, and outputs what is encrypted together with an EKB and a certificate to the other client for copying. In step S308, the CPU 21 writes the ICV computed in step S306 to a check list in the storage unit 28 in correspondence with the leaf key of the other client and the license ID.

[0228] If in step S303 the check-out count N1 is not judged smaller than (e.g., found equal to) the maximum check-out count N2, that means the maximum permissible check-out count has been exhausted so that the license can no longer be checked out. In that case, the CPU 21 goes to step S309 for error handling. The check-out process will be terminated unaccomplished.

[0229] Described below with reference to the flowchart of Fig. 43 is how a client has a license checked out from another client. This process takes place in conjunction with the check-out process of Fig. 42.

[0230] In step S321, the CPU 21 of the client 1 (of the check-out destination) transmits the leaf key of this client to another client (i.e., the license check-out source client). The leaf key is stored by the other client in correspondence with the license ID (in step S304).

[0231] In step S322, the CPU 21 receives from the other client 1 the encrypted license and ICV together with the EKB and certificate. The license, ICV, EKB, and certificate were transmitted earlier by the other client in step S307 of Fig. 42.

[0232] In step S323, the CPU 21 stores into the storage unit 28 the license, ICV, EKB, and certificate received in step S322.

[0233] The client 1 has the license checked out thereto from the other client in the manner described above. Thereafter, the client 1 reproduces the content corresponding to the checked-out license by carrying out the steps in the flowchart of Fig. 44.

[0234] In step S341, the CPU 21 of the client 1 computes the ICV of the content designated to be reproduced by the user through the input unit 26. In step S342, the CPU 21 decrypts the ICV in the storage unit 28 based on the public key included in the certificate.

[0235] In step S343, the CPU 21 judges whether or not the ICV computed in step S341 matches the ICV that was retrieved and decrypted in step S341. If the two values match, it means the license has not been tampered with. In that case, the CPU 21 goes to step S344 to reproduce the applicable content.

[0236] If in step S343 the two ICVs fail to match, that means the license may have been tampered with. In such a case, the CPU 21 goes to step S345 for error handling. Here, the content cannot be reproduced by use of the license in question.

[0237] Described below with reference to the flowchart of Fig. 45 is how a client has a previously checked-out license checked in from another client.

[0238] In step S361, the CPU 21 first acquires the leaf key of the other client (i.e., the client about to check in the license) and the ID of the license to be checked in. In step S362, the CPU 21 judges whether or not the target license whose ID was acquired in step S361 is a license previously checked out from this client to the other client. The judgment is made based on the ICV, leaf key and license ID stored in step S308 of Fig. 42. More specifically, a check is made to see whether or not the leaf key, license ID and ICV acquired in step S361 are held in the check-out list. If the leaf key, license ID and ICV are judged retained in the check-out list, that means the license in question has indeed been checked out by this client to the other client.

[0239] If the result of the check in step S362 is affirmative, then the CPU 21 goes to step S363 requesting the other client to delete the license, EKB and certificate involved. Given the request, the other client deletes the license, EKB and certificate as will be described later (in step S383 of Fig. 46).

[0240] In step S364, the CPU 21 decrements by 1 the check-out count N1 of the license in question. This is done to reflect the fact that a previously checked-out license is now returned (i.e., checked in).

[0241] In step S365, the CPU 21 determines whether or not this client has any other license still checked out to the other client. If there is no such license, step S366 is reached in which the CPU 21 deletes from the check-out list the record of the other client as a possible client for subsequent check-in. If in step S365 any other license is judged still checked out to the other client, the other client may subsequently request another check-in session and thus step S366 is skipped.

[0242] If in step S362 the license in question is not judged to be one previously checked out to the other client, then the CPU 21 goes to step S367 for error handling. In this case, the license in question is not subject to control by this client and the check-in process will not take place.

[0243] If the user has illegally copied the license, the stored ICV becomes different from the ICV computed on the basis of the license acquired in step S361. In that case, the check-in process will end unaccomplished.

[0244] Fig. 46 shows steps performed by a client having its license checked in to another client. This process takes place in conjunction with the license check-in

[0258] Shown on the right-hand side of Fig. 49 are steps for decrypting the delivered EKB and encrypted data. The devices 0, 1 and 2 first acquire the renewed node key K(t)00 by decrypting the received EKB using their own leaf keys or node keys. The renewed node key K(t)00 thus acquired is then used in a decrypting process to obtain the ICV generation key Kicv.

[0259] The devices 4, 5, 6, etc., in other groups shown in Fig. 12 may receive the same data (i.e., EKB) but are incapable of acquiring the renewed node key K(t)00 from the received data using their own leaf keys or node keys. Similarly, the revoked device 3 cannot obtain the renewed node key K(t)00 using its own leaf key or node key. Only the devices with legitimate rights are capable of decrypting the ICV generation value for their use.

[0260] Where the ICV generation key is delivered by use of the EKB as described above, it is possible to implement a scheme whereby the ICV generation key is delivered in a way securely decryptable only by those entitled to receive the key with a minimum of data amount involved.

[0261] The use of the integrity check value (ICV) for licenses makes it possible to eliminate illegal copy of EKBs and encrypted licenses. Illustratively, as shown in Fig. 50A, suppose that licenses L1 and L2 are stored on a storage medium 1 together with EKBs for allowing the licenses to be acquired and that what is stored on the storage medium 1 is copied entirely to a storage medium 2. In that case, with the EKBs and licenses copied onto the storage medium 2, the copied licenses can be used by any device capable of decrypting the EKBs.

[0262] In the example of Fig. 50B, the licenses held legitimately on a given storage medium are furnished with a corresponding integrity check value ICV(L1, L2). The value ICV(L1, L2) denotes an ICV given as

ICV = hash(Kicv, L1, L2)

which is an integrity check value computed by having a hash function applied to the licenses L1 and L2. In the example of Fig. 50B, the storage medium 1 legitimately contains the licenses L1 and L2 together with the integrity check value ICV(L1, L2) generated based on the two licenses. The storage medium 2 legitimately contains the license L1 along with an integrity check value ICV (L1) generated based on the license L1.

[0263] In the case of Fig. 50B, suppose that the EKB and the license 2 held on the storage medium 1 are copied to the storage medium 2 and that a license check value is generated anew for the storage medium 2. In that case, the integrity check value ICV(L1, L2) is generated which differs from Kicv(L1) retained on the storage medium 2. This reveals tampering with or illegal copy of the license that has been written to the storage medium. A device about to reproduce data from the storage medium carries out an ICV check before a data-reproducing step to determine whether or not there is a match between the generated ICV and the stored ICV. In case of a mismatch, the device will not reproduce data from the storage medium. This prevents reproduction of any illegally copied license.

[0264] In order to enhance security further, it is possible to generate the integrity check value (ICV) for each license on the basis of data including a renewal counter. More specifically, the ICV is computed as

ICV = hash(Kicv, counter+1, L1, L2, ...)

where the counter (counter+1) is established as a value that is incremented by 1 every time the ICV is renewed. The counter value needs to be stored in a secure memory.

[0265] Where the ICV for a license cannot be held on the same storage medium as the license in question, that ICV may be held on a storage medium separate from that of the license.

[0266] Illustratively, if a license is placed onto a read-only medium, an MO, or like storage medium that is not copy-protected, then putting the corresponding ICV on the same medium may prompt an unscrupulous user illegally to renew the ICV compromising its integrity. Such an eventuality is circumvented by keeping the ICVs on a secure storage medium in the host machine so that they are retrieved as needed for license copy control (e. g., check-in, check-out, move). This scheme provides securer ICV control measures and more elaborate license tampering checks.

[0267] The scheme above is typically implemented as shown in Fig. 51. In the example of Fig. 51, licenses 1, 2 and 3 are held on a storage medium 2201 such as a read-only medium, an MO or other storage medium that is not copy-protected. The ICV regarding these licenses is retained on a secure storage medium 2202 in the host machine that cannot be accessed freely by users. This arrangement prevents dishonest users from illegally renewing the integrity check value (ICV). Each device loaded with the storage medium 2201 requests the host machine such a PC or a server to perform ICV checks to determine whether or not data reproduction from the loaded storage medium is permitted. This effectively prevents illegal copying of or tampering with any license.

[0268] The clients to which this invention applies include not only so-called personal computers but also PDAs (personal digital assistants), mobile telephones and game consoles.

[0269] The series of steps described above may be executed either by hardware or by software. For software-based processing to take place, programs constituting the software may be either incorporated beforehand in dedicated hardware of a computer or installed upon use over a network or from a suitable program storage medium into a general-purpose personal computer or like equipment capable of executing diverse functions.

[0270] As shown in Fig. 2, the program storage medium is offered to users apart from computers not only as a package medium constituted by the magnetic disc 41 (including floppy discs), optical disc 42 (including CD-ROM (compact disc-read only memory) and DVD (digital versatile disc)), magneto-optical disc 43 (includ-

acquiring a certificate of a device subject to grouping;

authenticating said certificate acquired in said acquiring step;

encrypting a key for encrypting an object to be grouped, by use of a public key of said certificate authenticated in said authenticating step; and

providing said key which is encrypted in said encrypting step and which is used to encrypt said object to be grouped.

FIG. 2

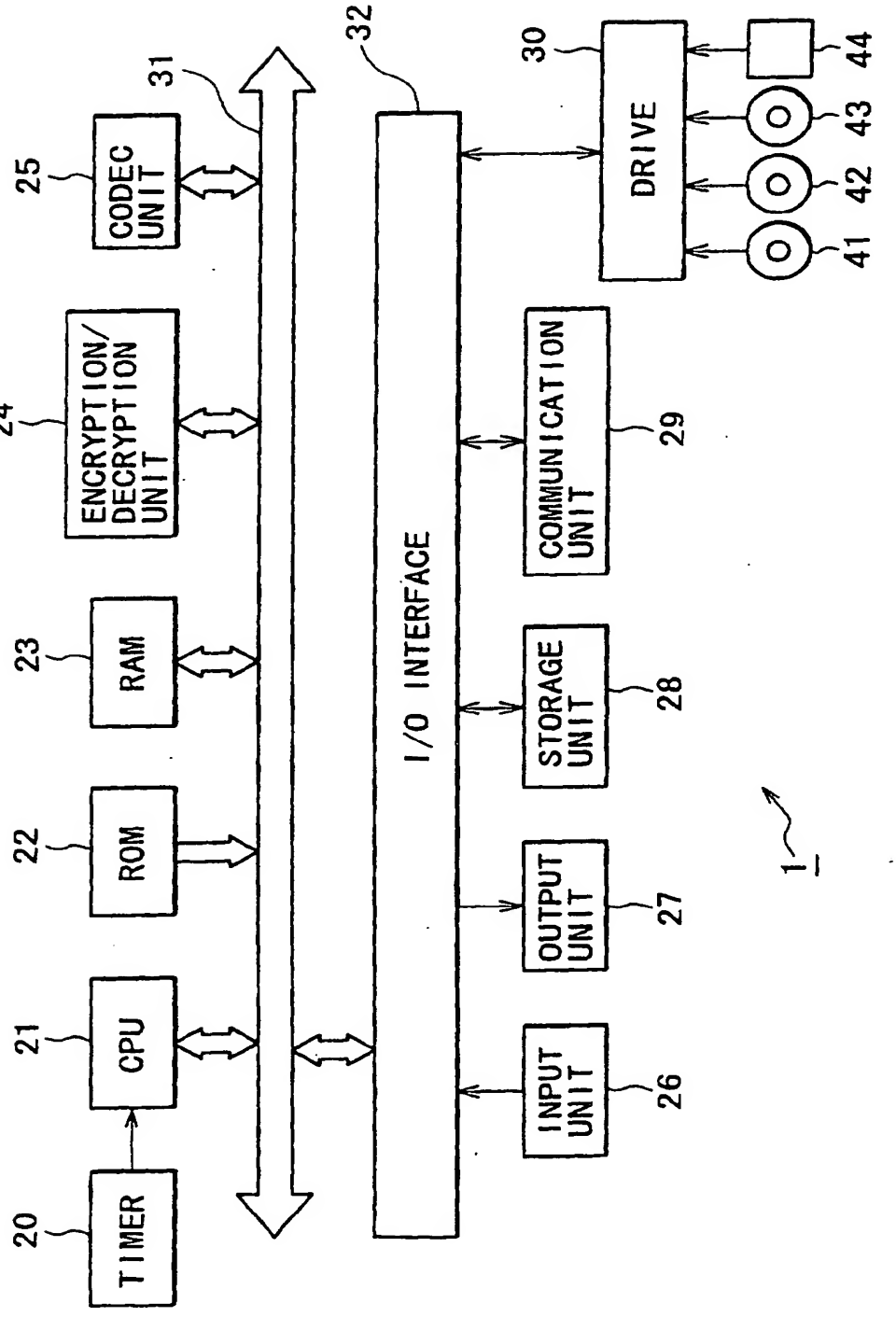

20
TIMER
21
CPU
22
ROM
23
RAM
24
ENCRYPTION/
DECRYPTION
UNIT
25
CODEC
UNIT
31
32
I/O INTERFACE
INPUT
UNIT
26
OUTPUT
UNIT
27
STORAGE
UNIT
28
COMMUNICATION
UNIT
29
30
DRIVE
41
42
43
44
1

FIG. 4

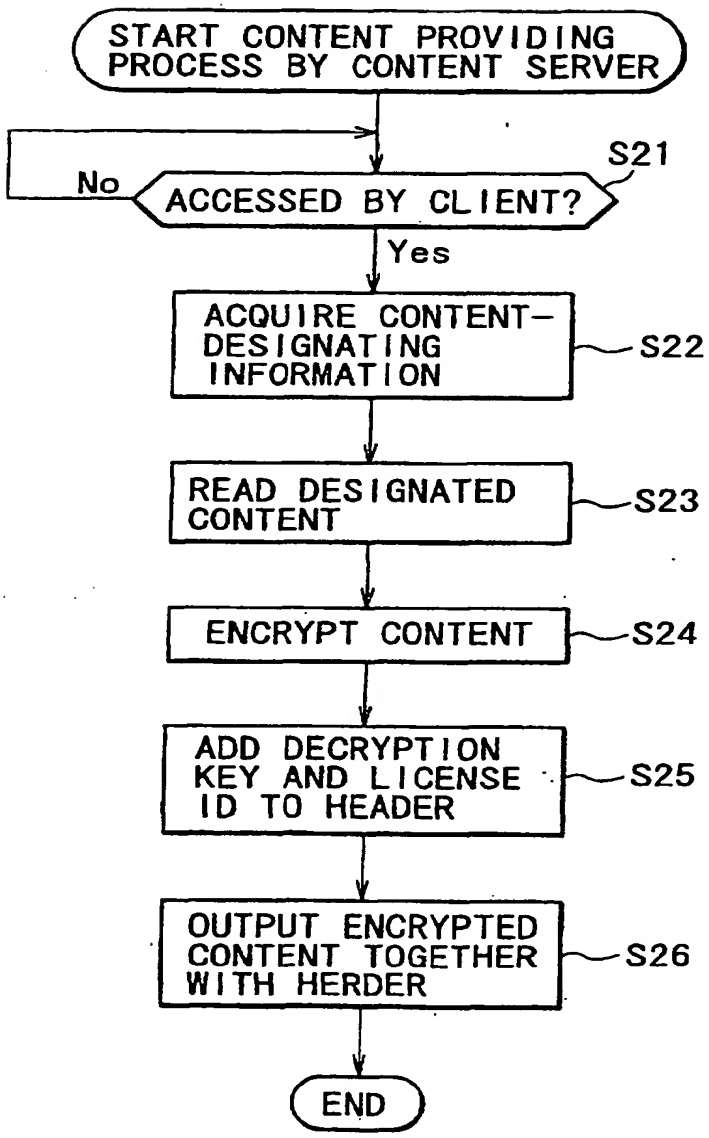

START CONTENT PROVIDING PROCESS BY CONTENT SERVER
S21
No
ACCESSED BY CLIENT?
Yes
ACQUIRE CONTENT-DESIGNATING INFORMATION
S22
READ DESIGNATED CONTENT
S23
ENCRYPT CONTENT
S24
ADD DECRYPTION KEY AND LICENSE ID TO HEADER
S25
OUTPUT ENCRYPTED CONTENT TOGETHER WITH HERDER
S26
END

FIG. 6

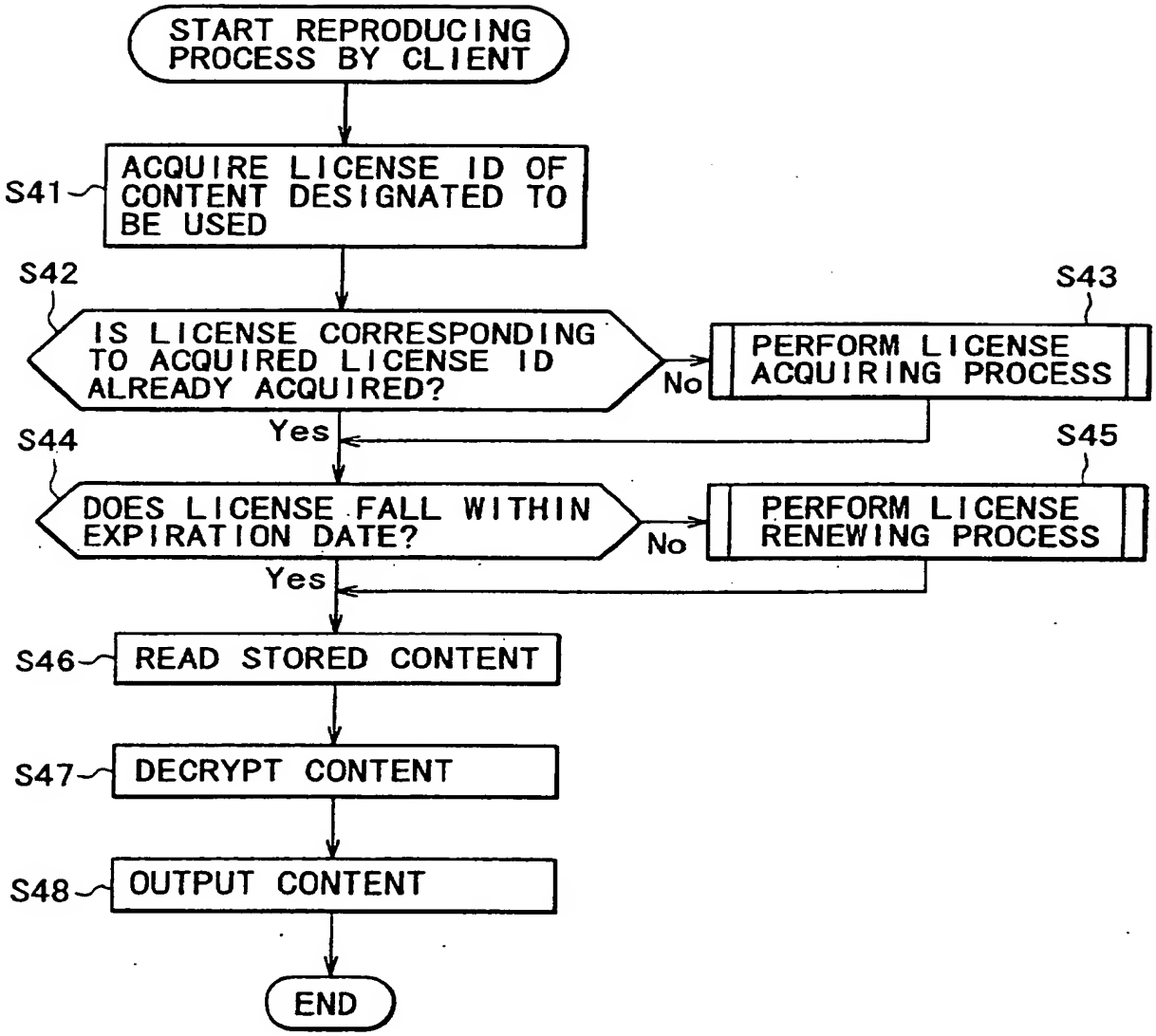

START REPRODUCING PROCESS BY CLIENT
S41
ACQUIRE LICENSE ID OF CONTENT DESIGNATED TO BE USED
S42
IS LICENSE CORRESPONDING TO ACQUIRED LICENSE ID ALREADY ACQUIRED?
No
S43
PERFORM LICENSE ACQUIRING PROCESS
Yes
S44
DOES LICENSE FALL WITHIN EXPIRATION DATE?
No
S45
PERFORM LICENSE RENEWING PROCESS
Yes
S46
READ STORED CONTENT
S47
DECRYPT CONTENT
S48
OUTPUT CONTENT
END

FIG. 8

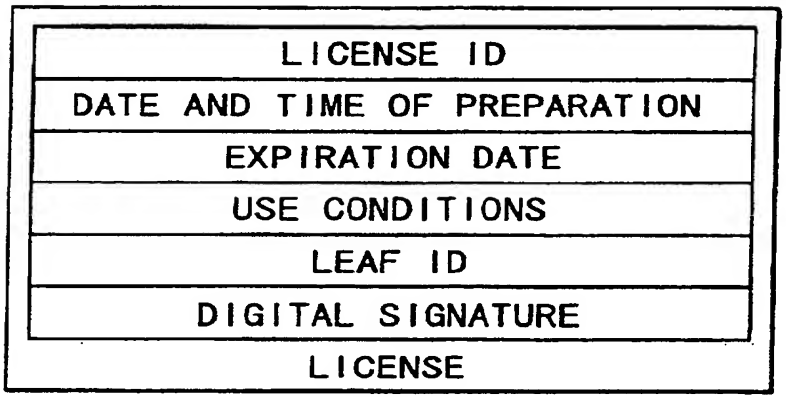

LICENSE ID
DATE AND TIME OF PREPARATION
EXPIRATION DATE
USE CONDITIONS
LEAF ID
DIGITAL SIGNATURE
LICENSE

FIG. 10

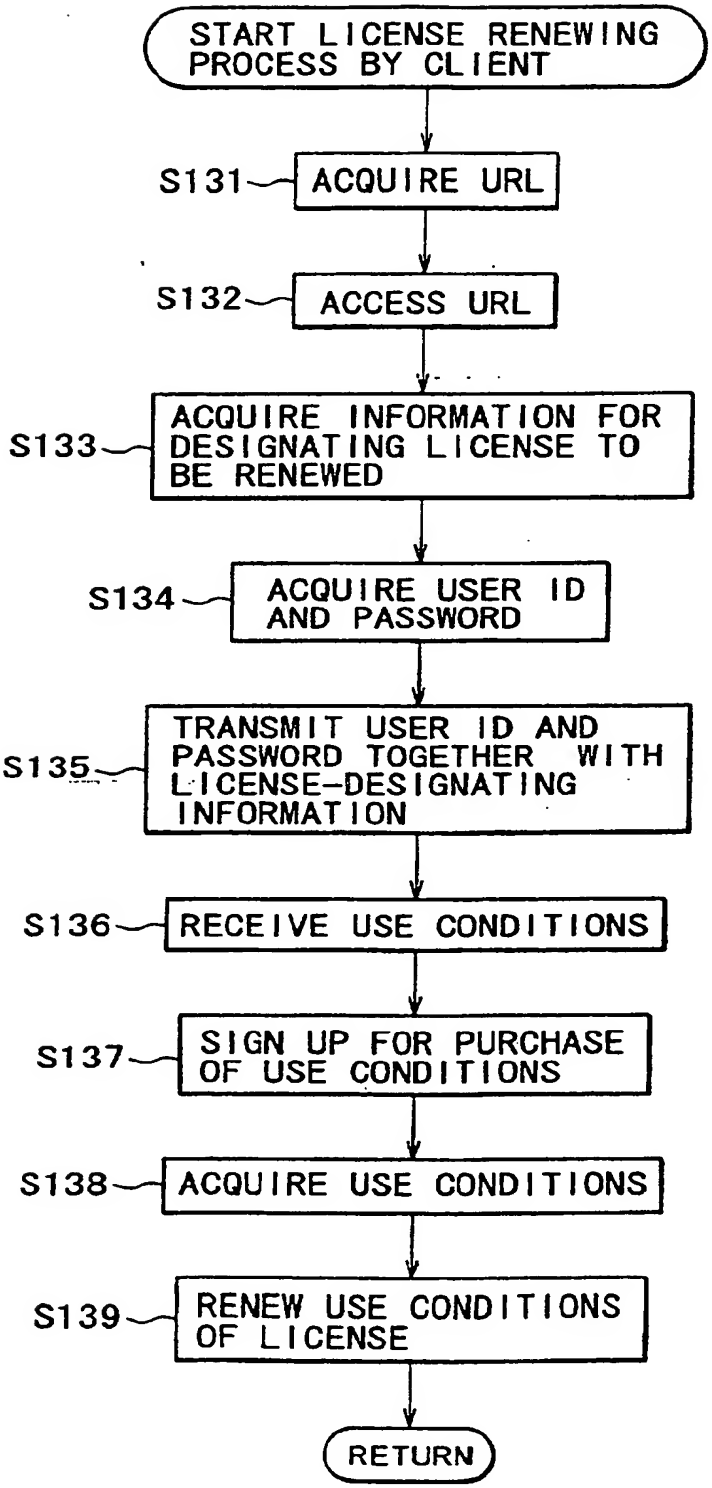

START LICENSE RENEWING PROCESS BY CLIENT
S131
ACQUIRE URL
S132
ACCESS URL
S133
ACQUIRE INFORMATION FOR DESIGNATING LICENSE TO BE RENEWED
S134
ACQUIRE USER ID AND PASSWORD
S135
TRANSMIT USER ID AND PASSWORD TOGETHER WITH LICENSE-DESIGNATING INFORMATION
S136
RECEIVE USE CONDITIONS
S137
SIGN UP FOR PURCHASE OF USE CONDITIONS
S138
ACQUIRE USE CONDITIONS
S139
RENEW USE CONDITIONS OF LICENSE
RETURN

FIG. 12

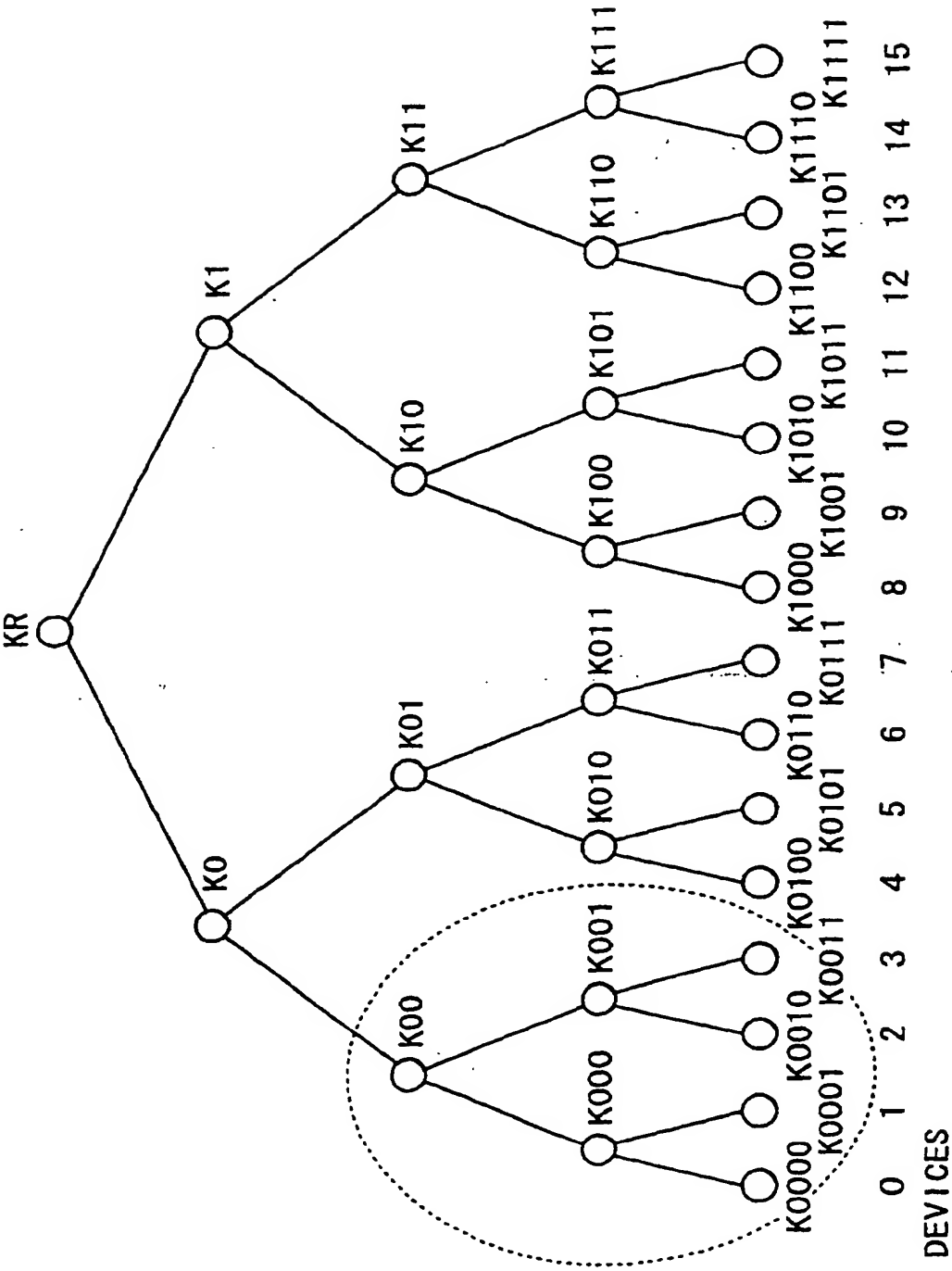

KR
K0
K1
K00
K01
K10
K11
K000
K001
K010
K011
K100
K101
K110
K111
K0000
K0001
K0010
K0011
K0100
K0101
K0110
K0111
K1000
K1001
K1010
K1011
K1100
K1101
K1110
K1111
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
DEVICES

FIG. 14

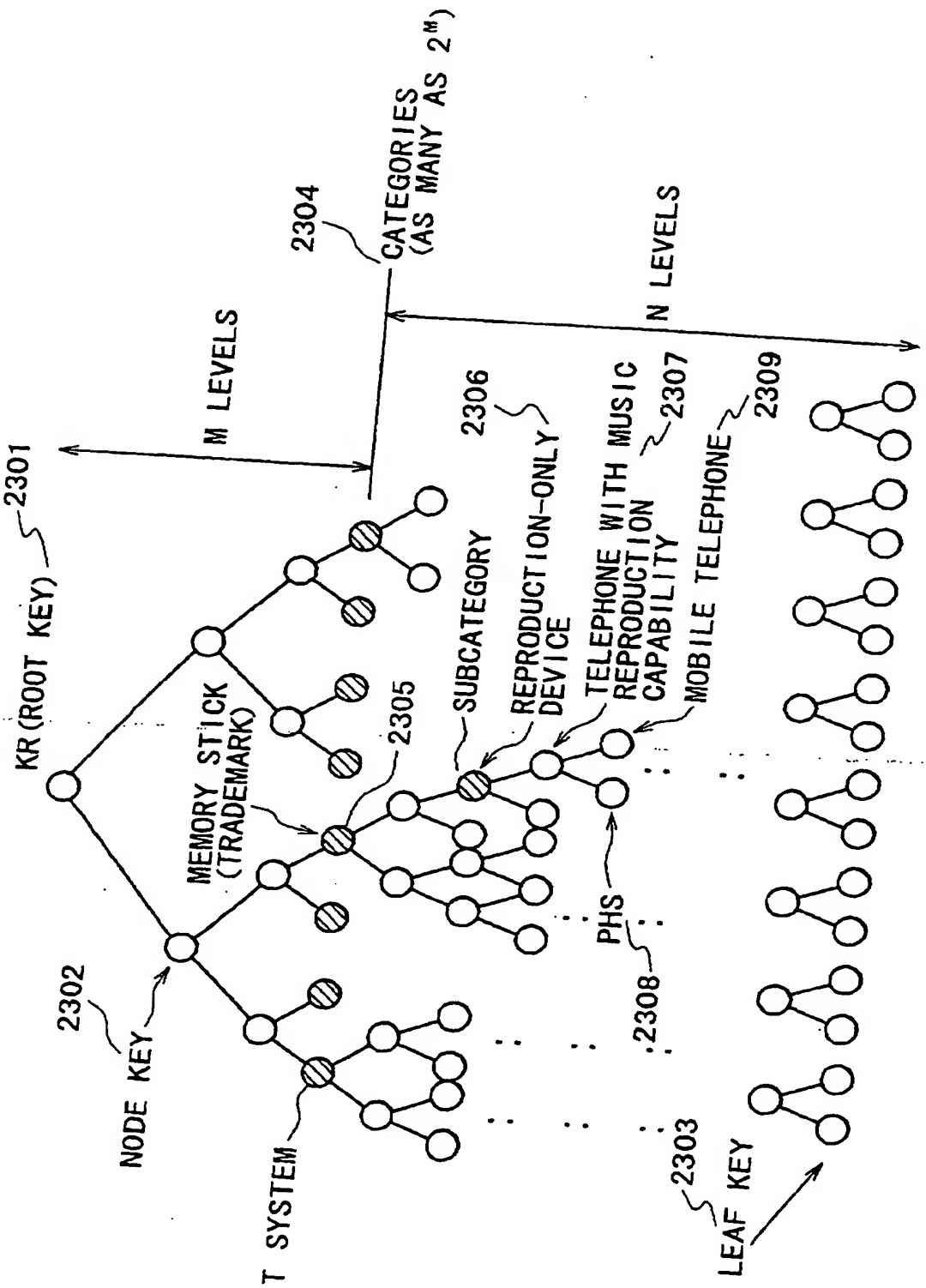

KR (ROOT KEY)
2301
M LEVELS
2304
CATEGORIES (AS MANY AS 2^M)
N LEVELS
2306
SUBCATEGORY
REPRODUCTION-ONLY DEVICE
TELEPHONE WITH MUSIC REPRODUCTION CAPABILITY
2307
MOBILE TELEPHONE
2309
MEMORY STICK (TRADEMARK)
2305
PHS
2308
2302
NODE KEY
T SYSTEM
2303
LEAF KEY

FIG. 16

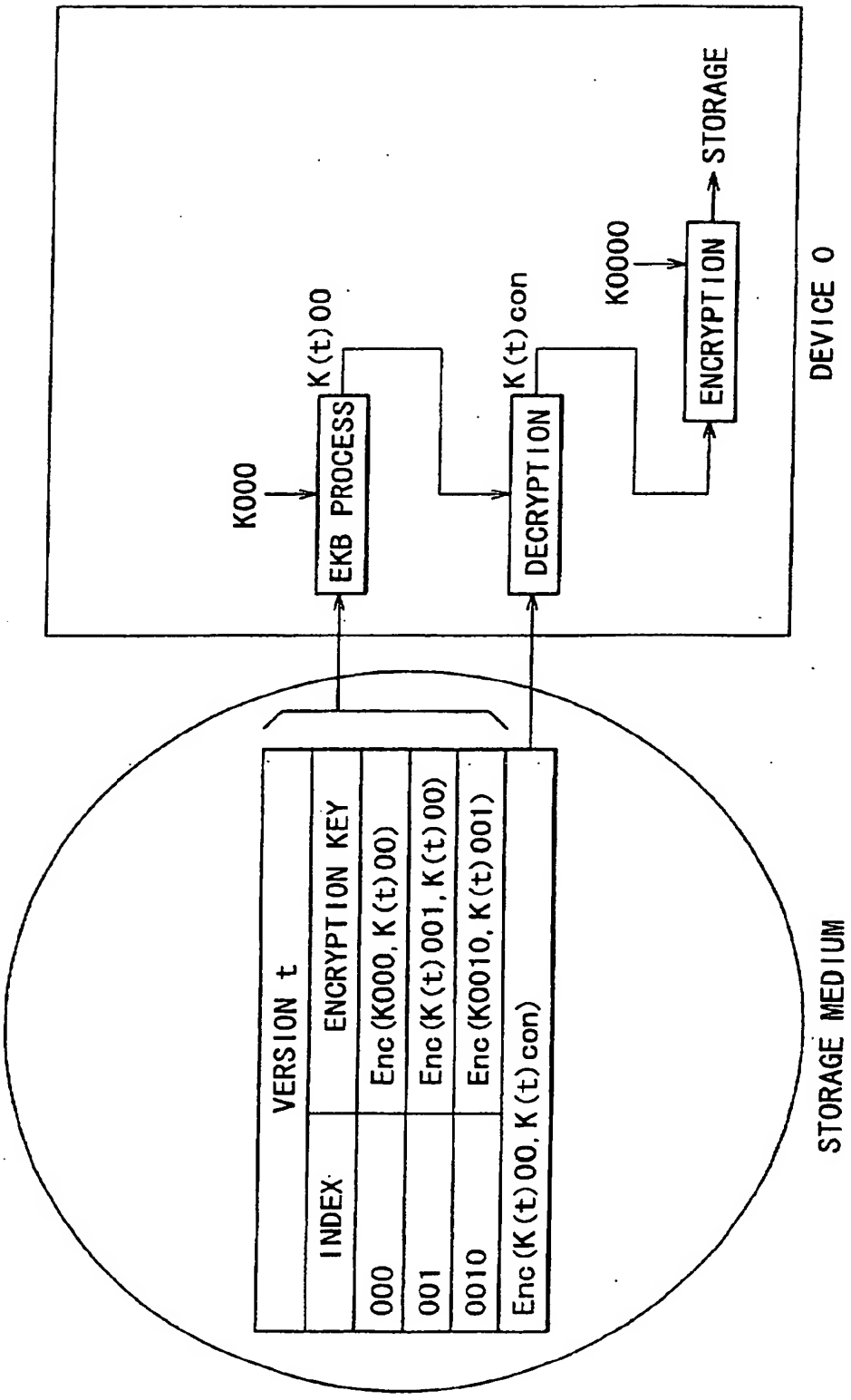

K000
EKB PROCESS
K(t)00
K(t)con
DECRYPTION
K0000
ENCRYPTION
STORAGE
DEVICE 0
VERSION t
INDEX
ENCRYPTION KEY
000
Enc(K000, K(t)00)
001
Enc(K(t)001, K(t)00)
0010
Enc(K0010, K(t)001)
Enc(K(t)00, K(t)con)
STORAGE MEDIUM

FIG. 18

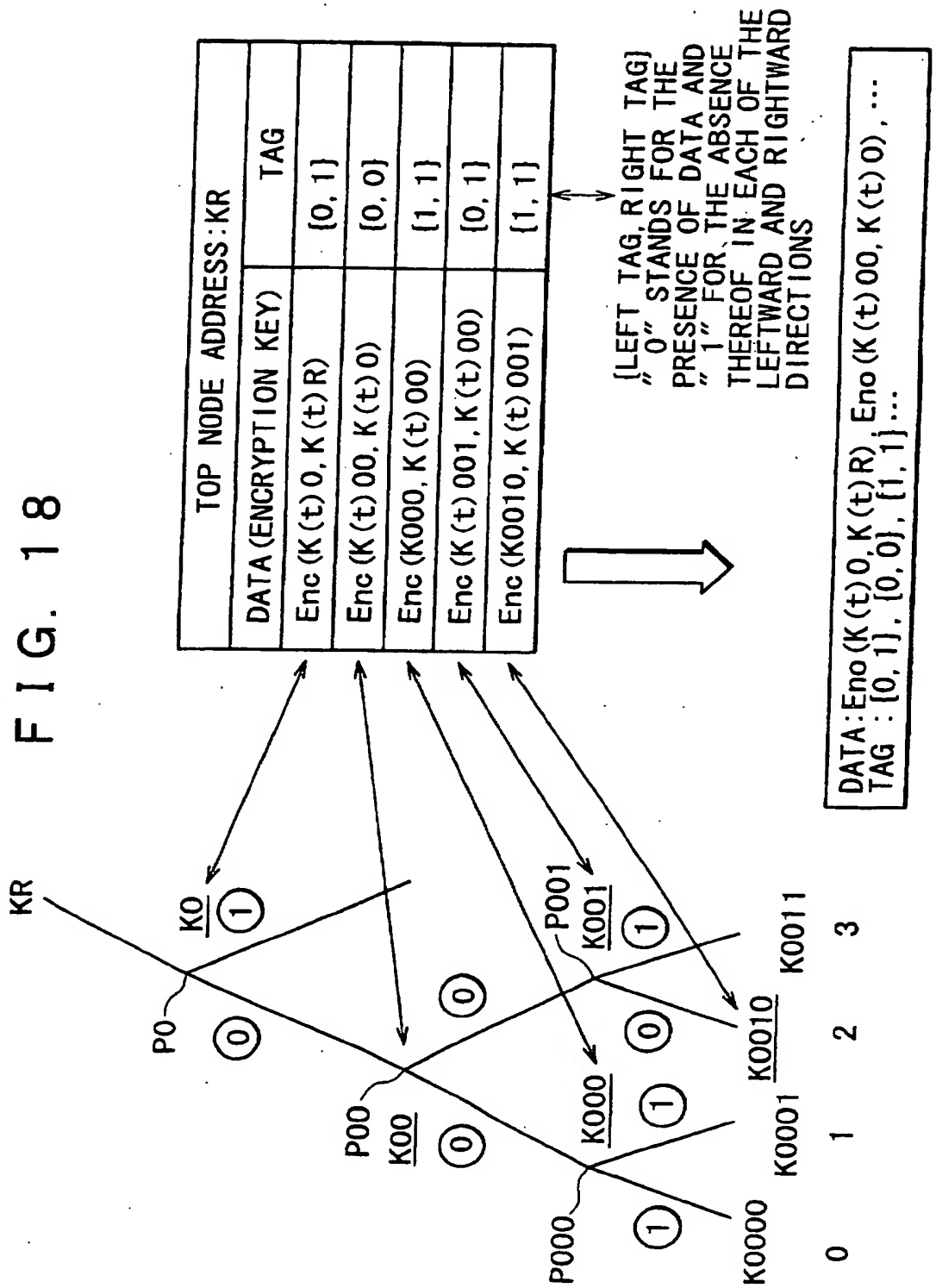

TOP NODE ADDRESS:KR
DATA (ENCRYPTION KEY)
TAG
Enc (K(t)0, K(t)R)
{0, 1}
Enc (K(t)00, K(t)0)
{0, 0}
Enc (K000, K(t)00)
{1, 1}
Enc (K(t)001, K(t)00)
{0, 1}
Enc (K0010, K(t)001)
{1, 1}
{LEFT TAG, RIGHT TAG}
"0" STANDS FOR THE PRESENCE OF DATA AND "1" FOR THE ABSENCE THEREOF IN EACH OF THE LEFTWARD AND RIGHTWARD DIRECTIONS
DATA: Eno (K(t)0, K(t)R), Eno (K(t)00, K(t)0), ...
TAG : {0, 1}, {0, 0}, {1, 1} ...
KR
K0
P0
K00
P00
K000
P000
K0000
P001
K001
K0010
K0011
K0001
0
1
2
3

FIG. 21

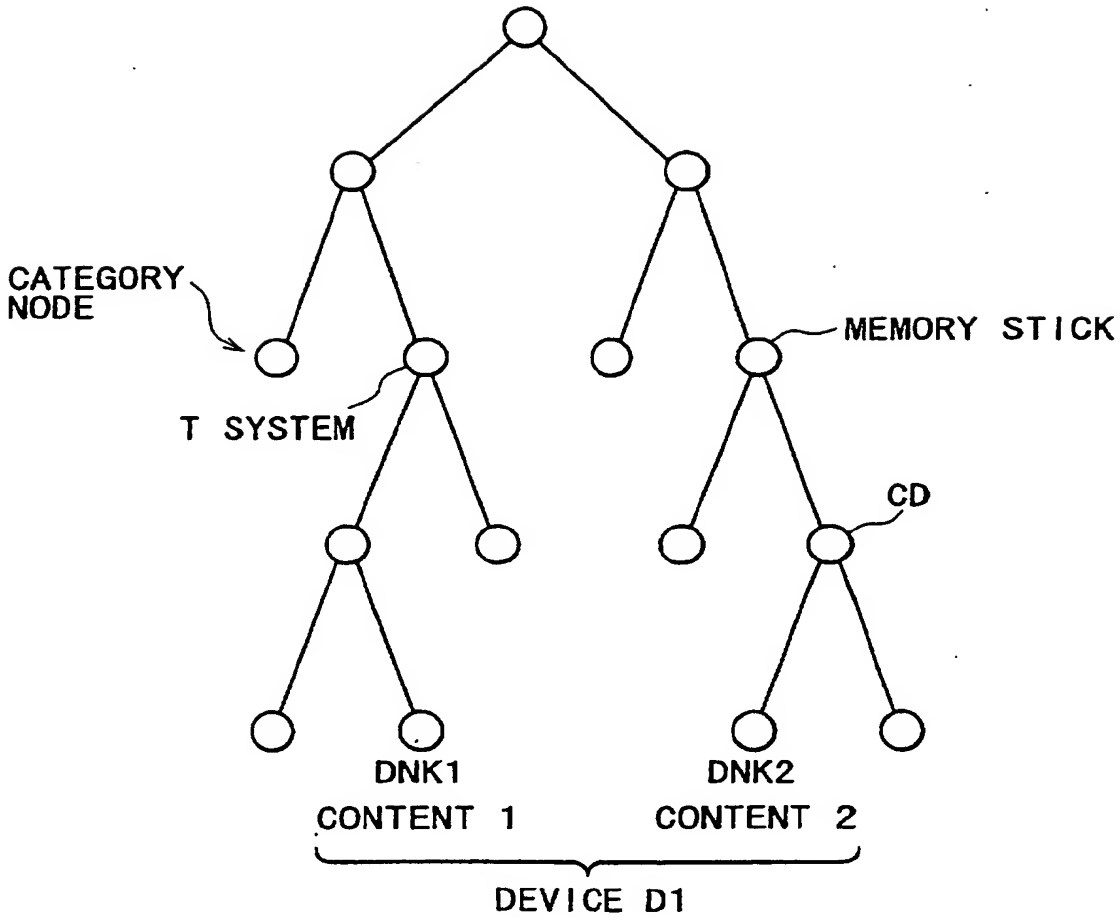

CATEGORY NODE
MEMORY STICK
T SYSTEM
CD
DNK1
DNK2
CONTENT 1
CONTENT 2
DEVICE D1

FIG. 23

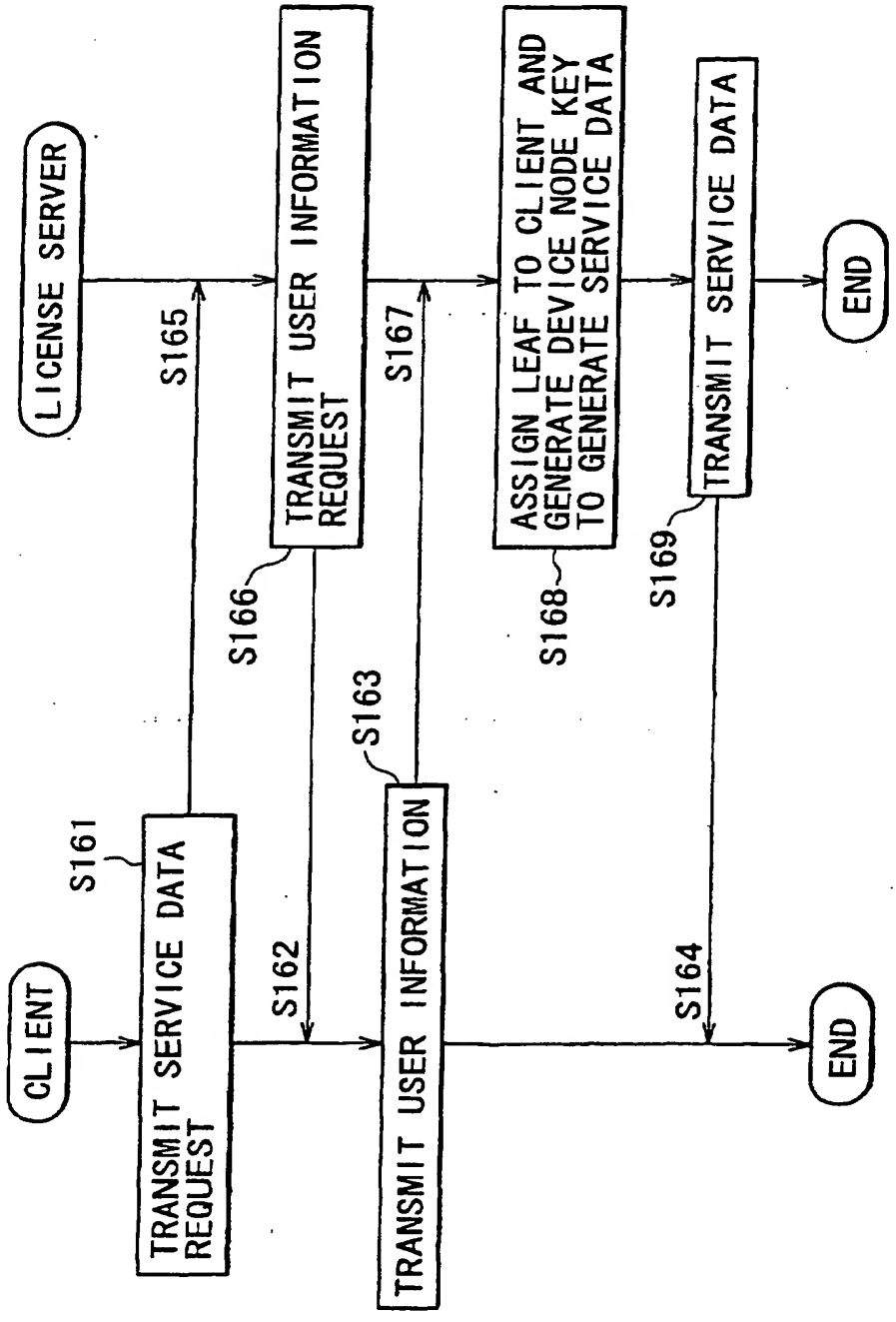

CLIENT
LICENSE SERVER
TRANSMIT SERVICE DATA REQUEST
TRANSMIT USER INFORMATION REQUEST
TRANSMIT USER INFORMATION
ASSIGN LEAF TO CLIENT AND GENERATE DEVICE NODE KEY TO GENERATE SERVICE DATA
TRANSMIT SERVICE DATA
END
END
S161
S162
S163
S164
S165
S166
S167
S168
S169

FIG. 25

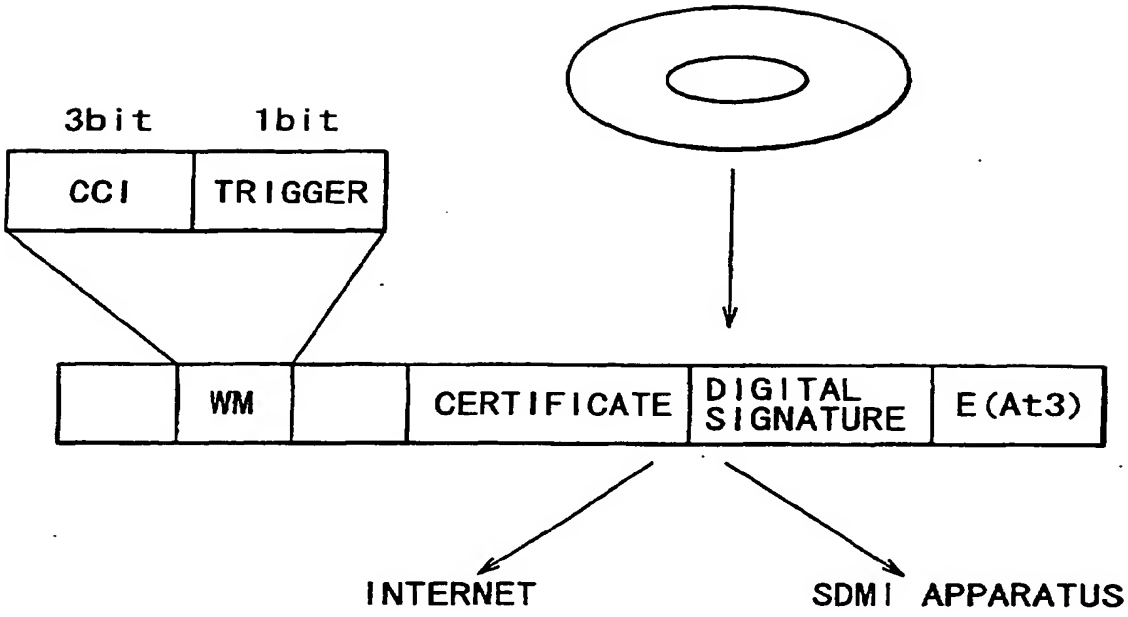

3bit
1bit
CCI
TRIGGER
WM
CERTIFICATE
DIGITAL SIGNATURE
E(At3)
INTERNET
SDMI APPARATUS

FIG. 26

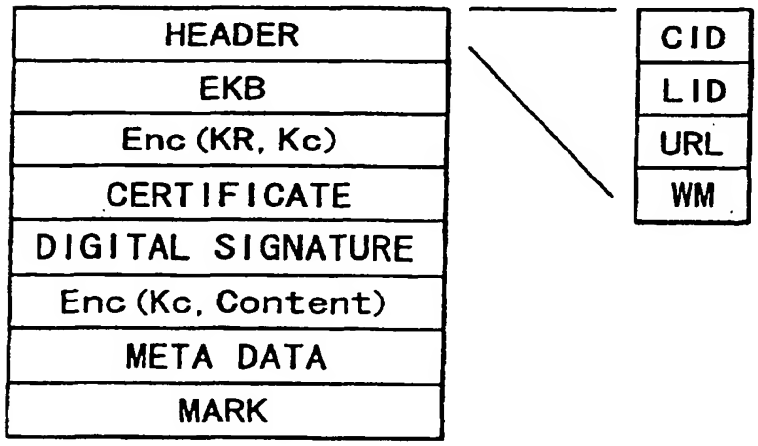

HEADER
EKB
Enc (KR, Kc)
CERTIFICATE
DIGITAL SIGNATURE
Enc (Kc, Content)
META DATA
MARK
CID
LID
URL
WM

FIG. 28

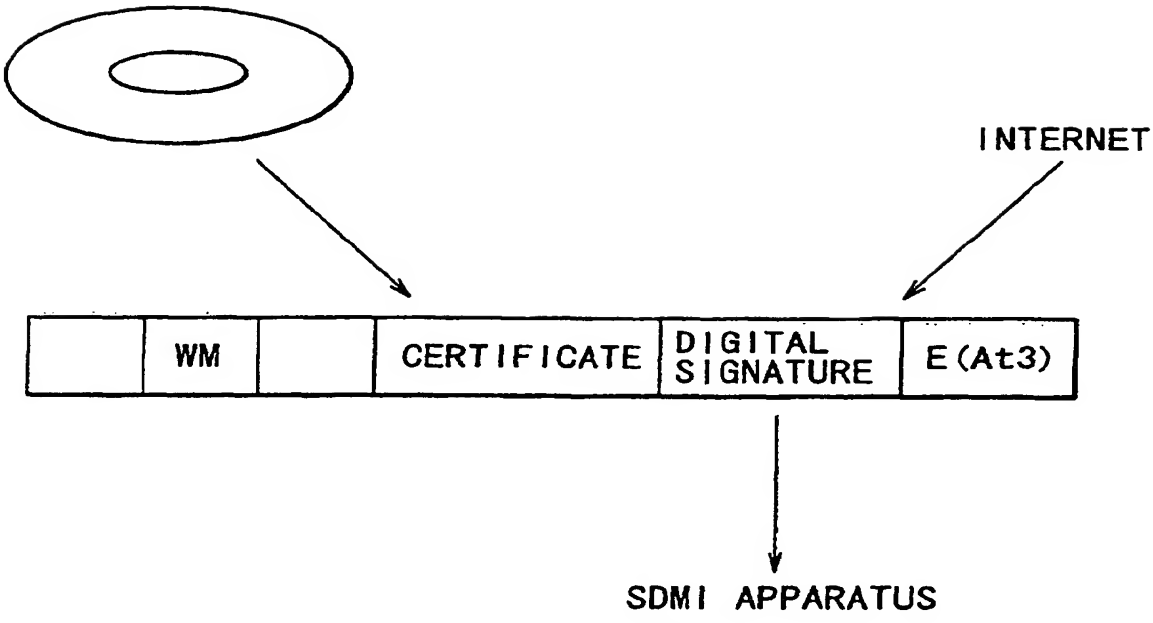

INTERNET
WM
CERTIFICATE
DIGITAL SIGNATURE
E(At3)
SDMI APPARATUS

FIG. 30

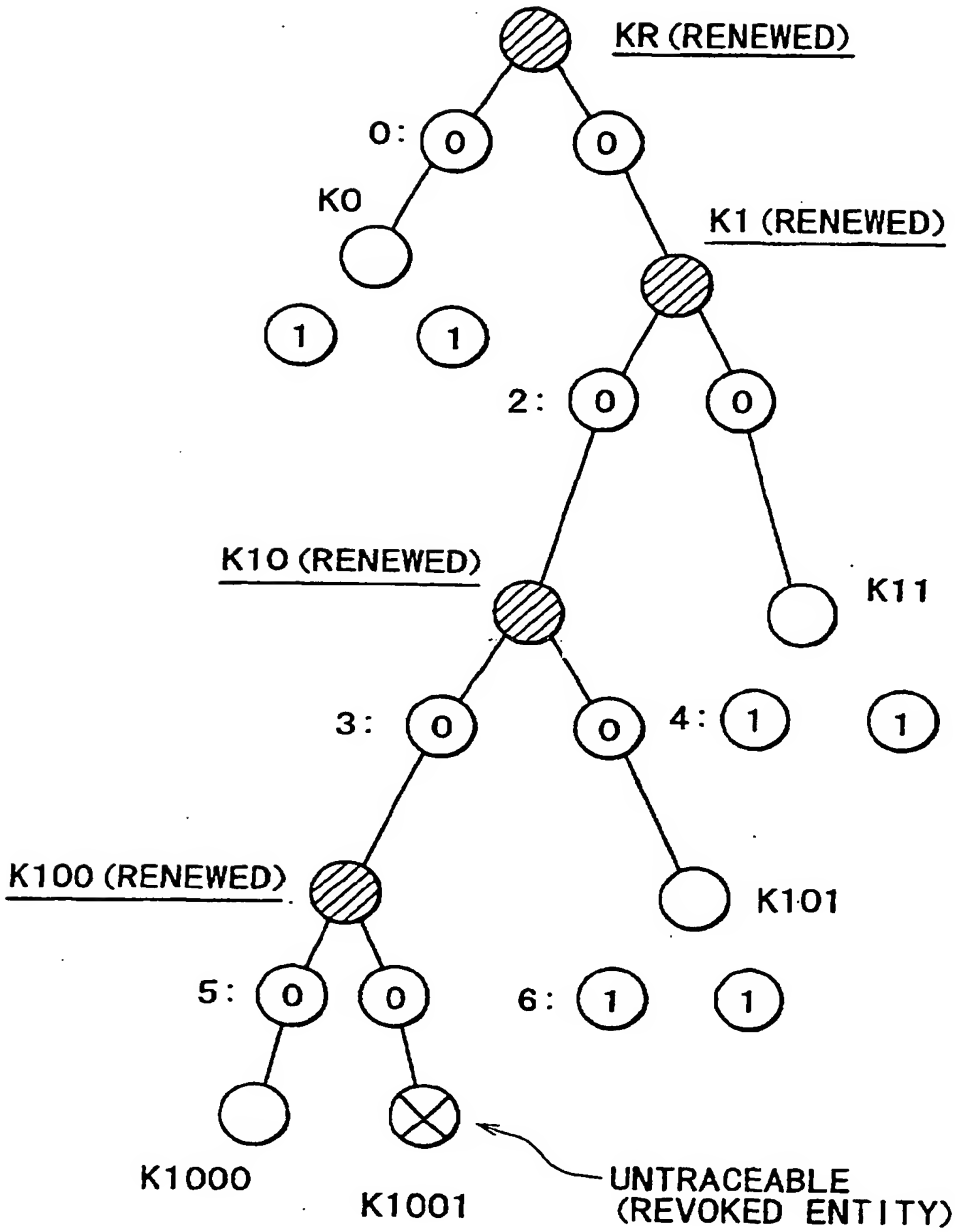

KR (RENEWED)
0:
0
0
K0
K1 (RENEWED)
1
1
2:
0
0
K10 (RENEWED)
K11
3:
0
0
4:
1
1
K100 (RENEWED)
K101
5:
0
0
6:
1
1
K1000
K1001
UNTRACEABLE
(REVOKED ENTITY)

FIG. 33

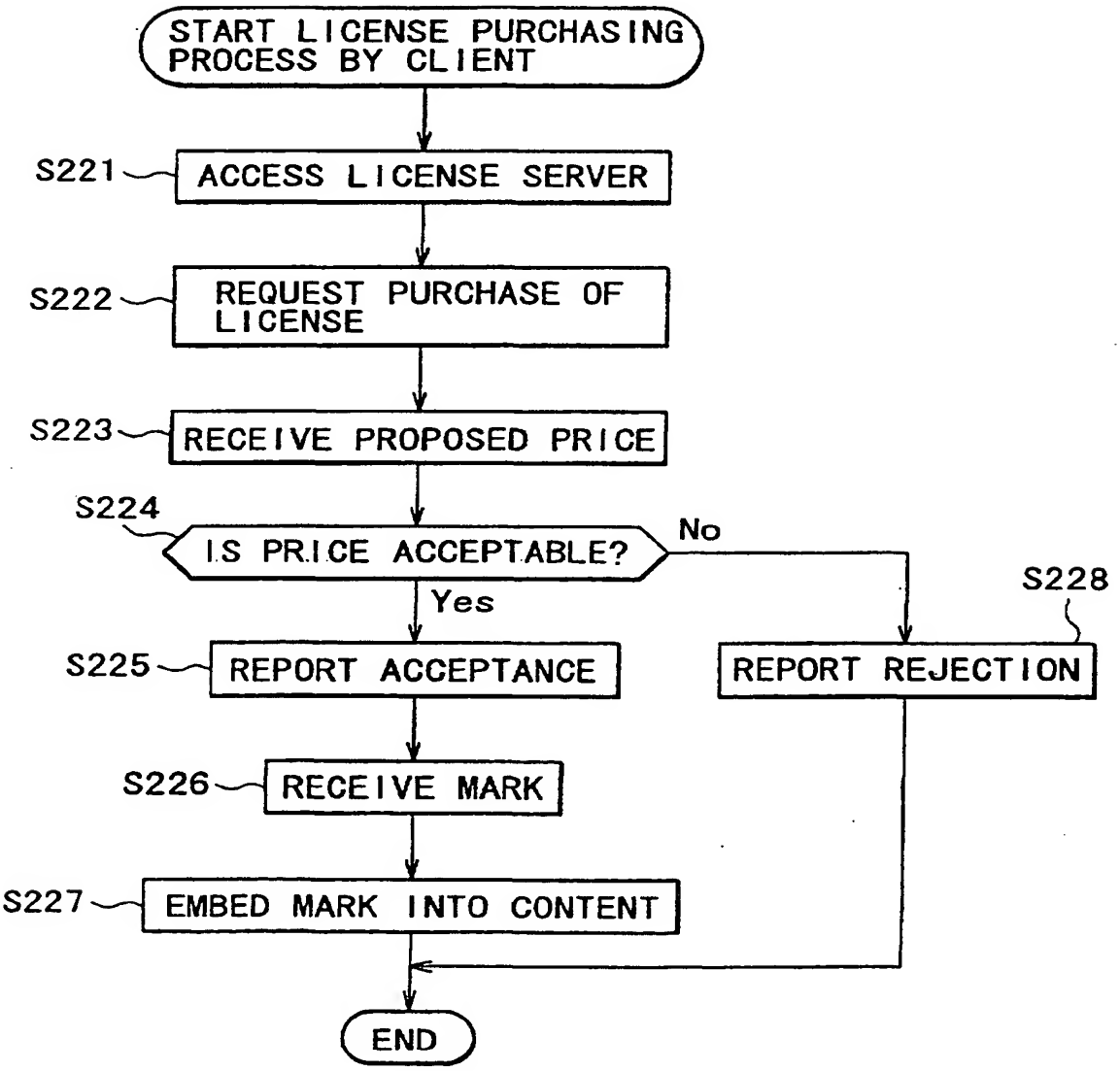

START LICENSE PURCHASING PROCESS BY CLIENT
S221
ACCESS LICENSE SERVER
S222
REQUEST PURCHASE OF LICENSE
S223
RECEIVE PROPOSED PRICE
S224
IS PRICE ACCEPTABLE?
No
Yes
S225
REPORT ACCEPTANCE
S226
RECEIVE MARK
S227
EMBED MARK INTO CONTENT
S228
REPORT REJECTION
END

FIG. 35

Mark={LeafID, Own, Sigs(LeafID, Own)}

FIG. 36

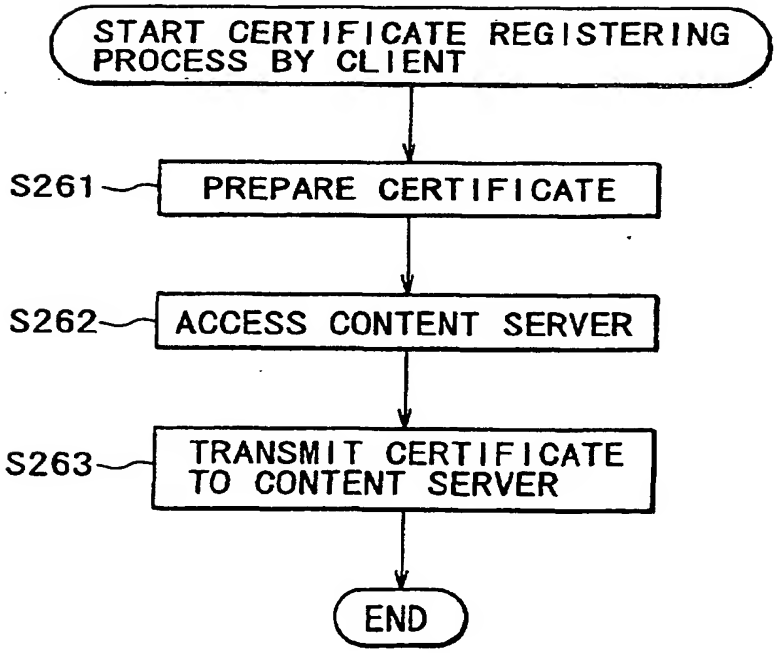


FIG. 39

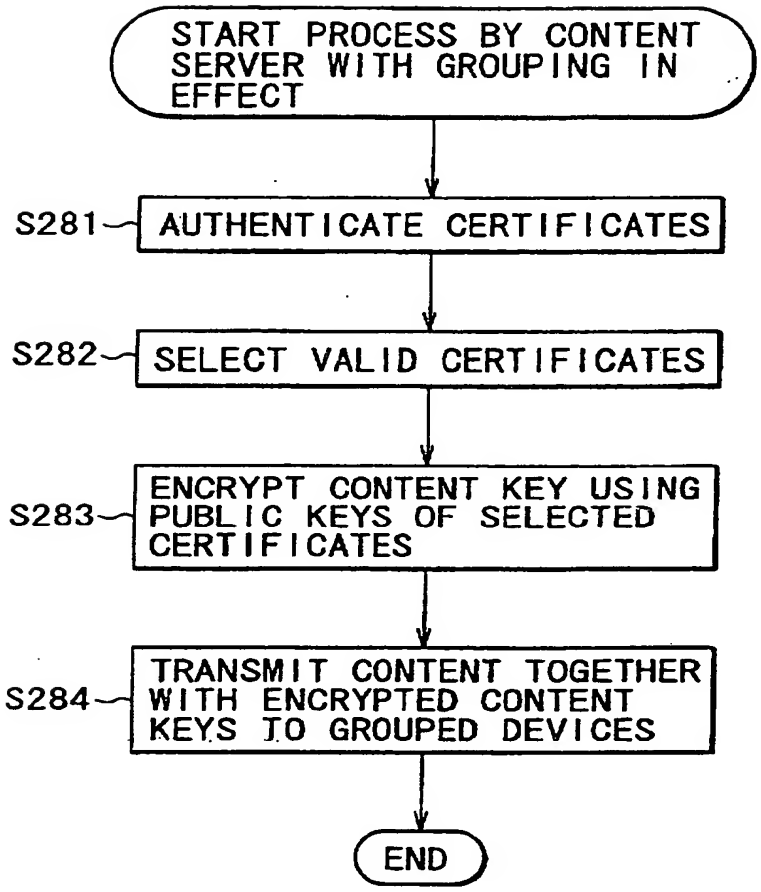

START PROCESS BY CONTENT SERVER WITH GROUPING IN EFFECT
S281
AUTHENTICATE CERTIFICATES
S282
SELECT VALID CERTIFICATES
S283
ENCRYPT CONTENT KEY USING PUBLIC KEYS OF SELECTED CERTIFICATES
S284
TRANSMIT CONTENT TOGETHER WITH ENCRYPTED CONTENT KEYS TO GROUPED DEVICES
END

FIG. 40

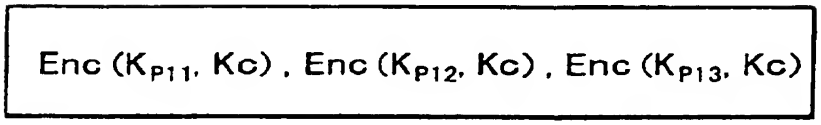

Enc (K_{P11}, Kc) , Enc (K_{P12}, Kc) , Enc (K_{P13}, Kc)

FIG. 42

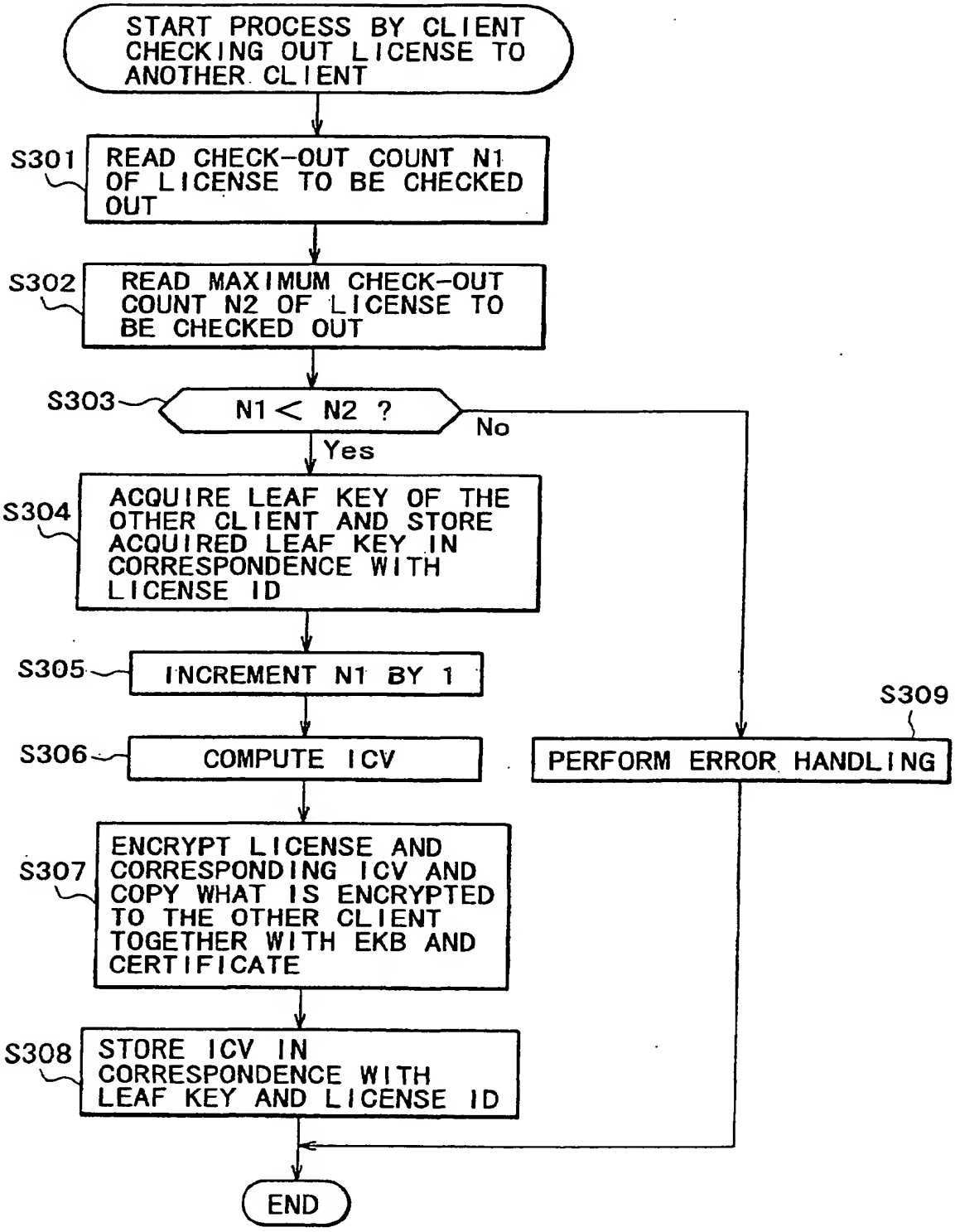

START PROCESS BY CLIENT CHECKING OUT LICENSE TO ANOTHER CLIENT
S301
READ CHECK-OUT COUNT N1 OF LICENSE TO BE CHECKED OUT
S302
READ MAXIMUM CHECK-OUT COUNT N2 OF LICENSE TO BE CHECKED OUT
S303
N1 < N2 ?
No
Yes
S304
ACQUIRE LEAF KEY OF THE OTHER CLIENT AND STORE ACQUIRED LEAF KEY IN CORRESPONDENCE WITH LICENSE ID
S305
INCREMENT N1 BY 1
S306
COMPUTE ICV
S307
ENCRYPT LICENSE AND CORRESPONDING ICV AND COPY WHAT IS ENCRYPTED TO THE OTHER CLIENT TOGETHER WITH EKB AND CERTIFICATE
S308
STORE ICV IN CORRESPONDENCE WITH LEAF KEY AND LICENSE ID
S309
PERFORM ERROR HANDLING
END

FIG. 44

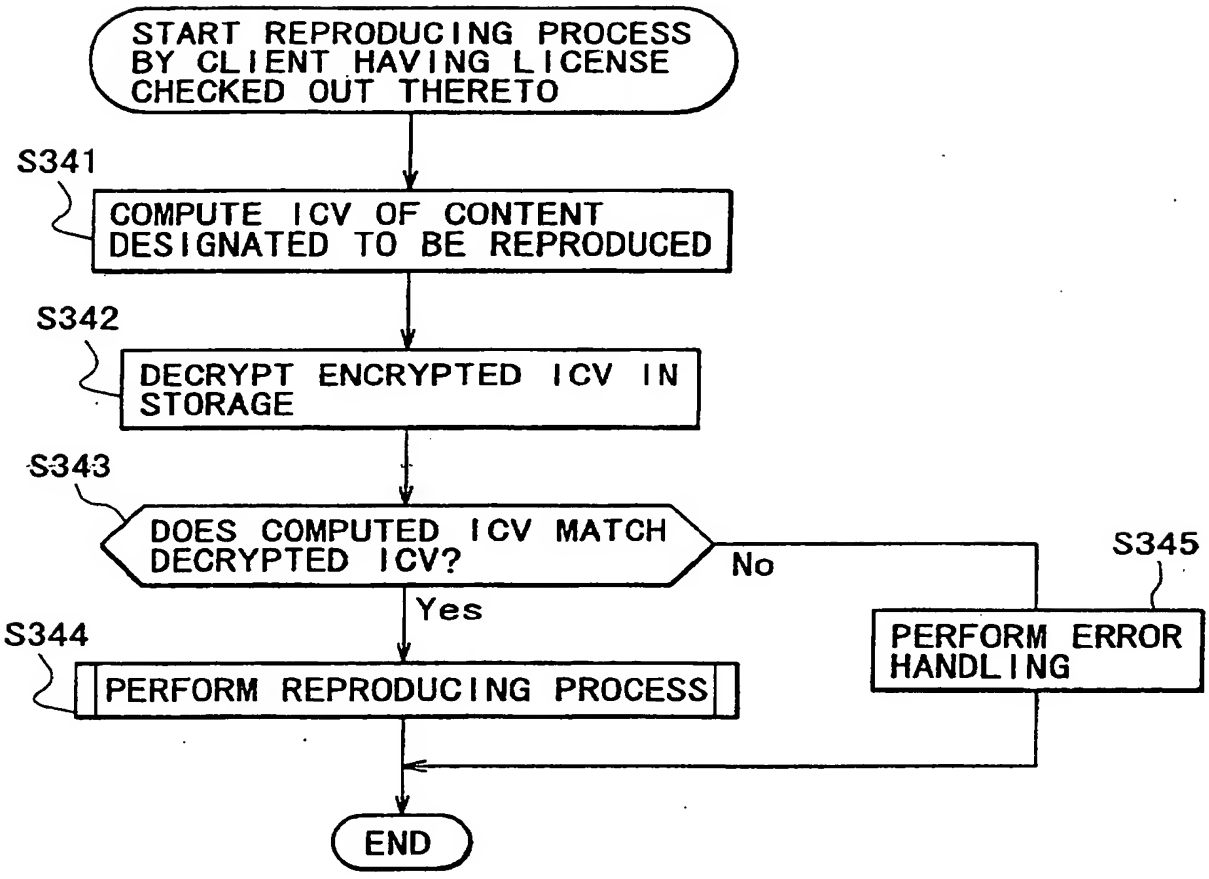

START REPRODUCING PROCESS BY CLIENT HAVING LICENSE CHECKED OUT THERETO
S341
COMPUTE ICV OF CONTENT DESIGNATED TO BE REPRODUCED
S342
DECRYPT ENCRYPTED ICV IN STORAGE
S343
DOES COMPUTED ICV MATCH DECRYPTED ICV?
No
Yes
S345
PERFORM ERROR HANDLING
S344
PERFORM REPRODUCING PROCESS
END

FIG. 46

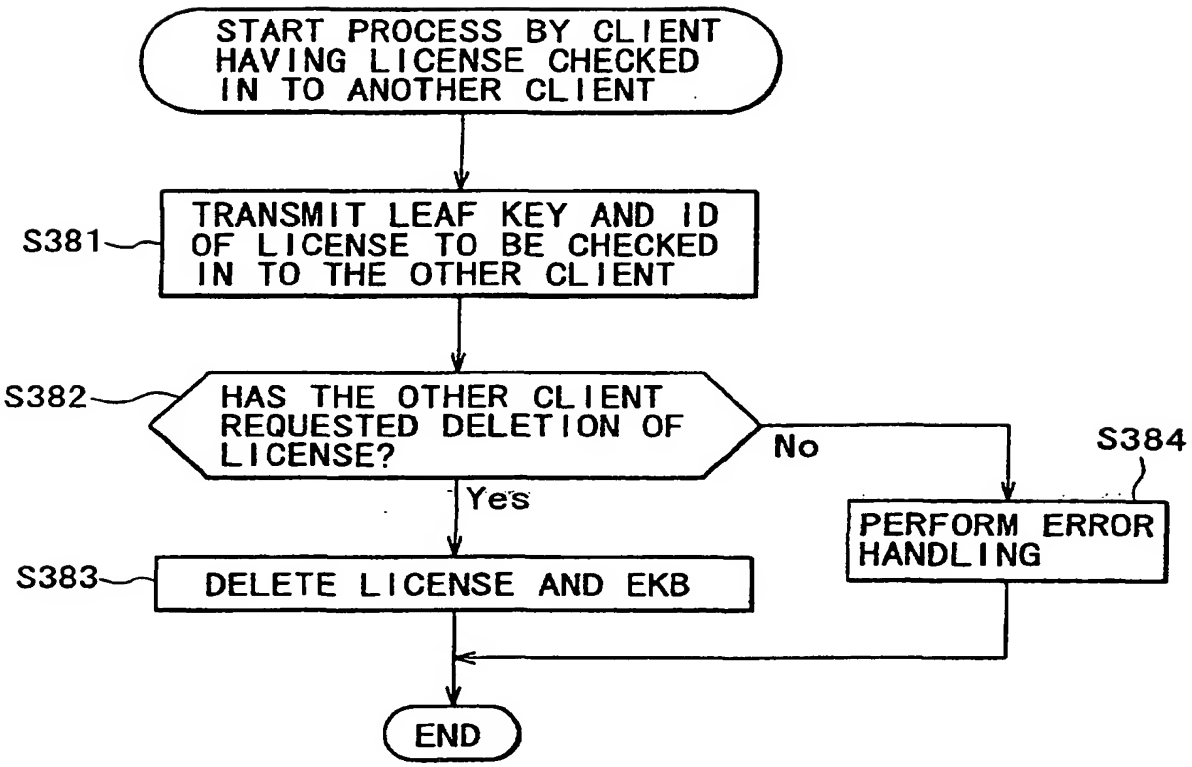

START PROCESS BY CLIENT HAVING LICENSE CHECKED IN TO ANOTHER CLIENT
S381
TRANSMIT LEAF KEY AND ID OF LICENSE TO BE CHECKED IN TO THE OTHER CLIENT
S382
HAS THE OTHER CLIENT REQUESTED DELETION OF LICENSE?
No
S384
PERFORM ERROR HANDLING
Yes
S383
DELETE LICENSE AND EKB
END

FIG. 48

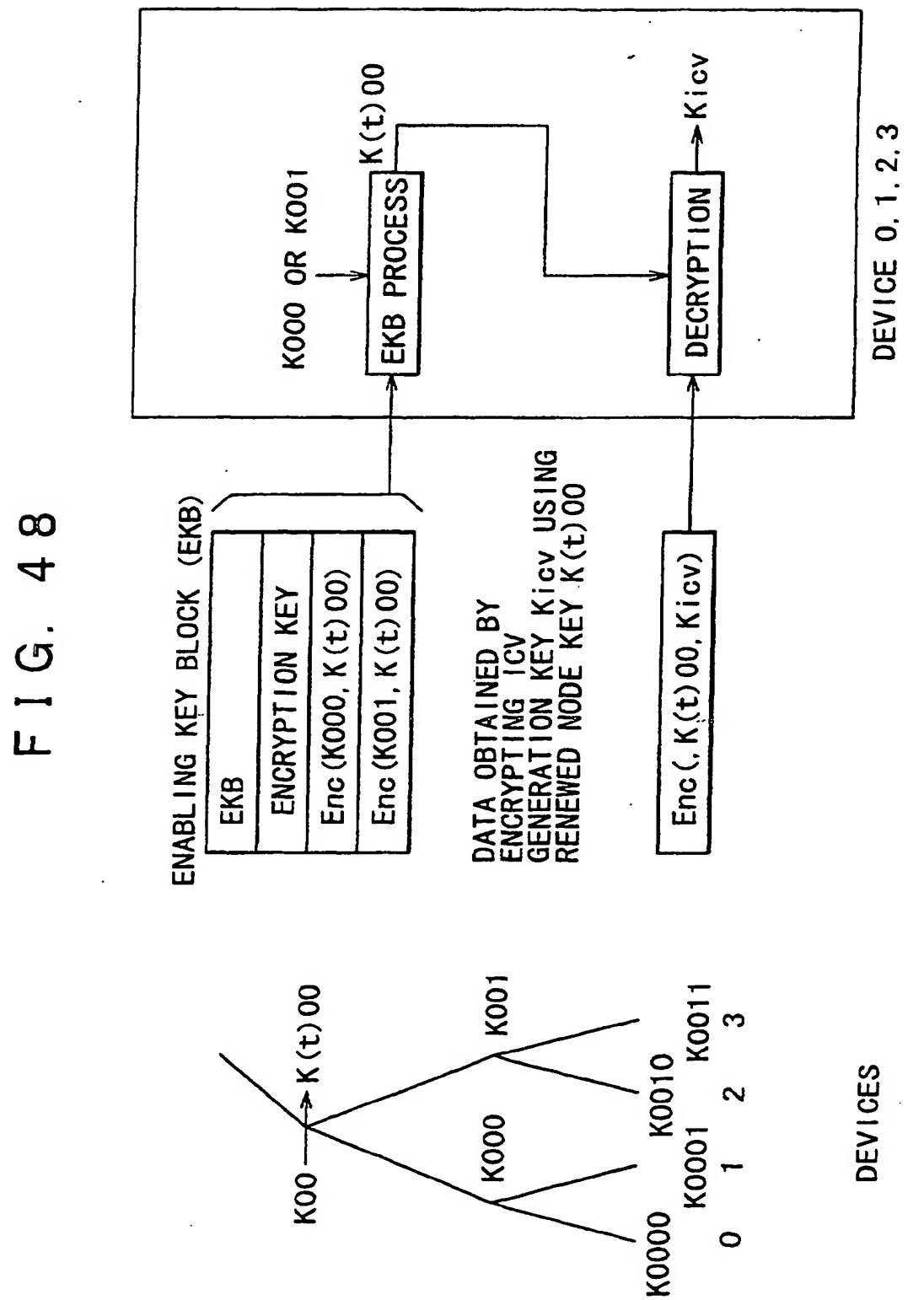

DEVICE 0, 1, 2, 3
K000 OR K001
EKB PROCESS
K(t)00
DECRYPTION
Kicv
ENABLING KEY BLOCK (EKB)
EKB
ENCRYPTION KEY
Enc(K000, K(t)00)
Enc(K001, K(t)00)
DATA OBTAINED BY ENCRYPTING ICV GENERATION KEY Kicv USING RENEWED NODE KEY K(t)00
Enc(, K(t)00, Kicv)
K00
K(t)00
K000
K001
K0000
K0001
K0010
K0011
0
1
2
3
DEVICES

FIG. 50A

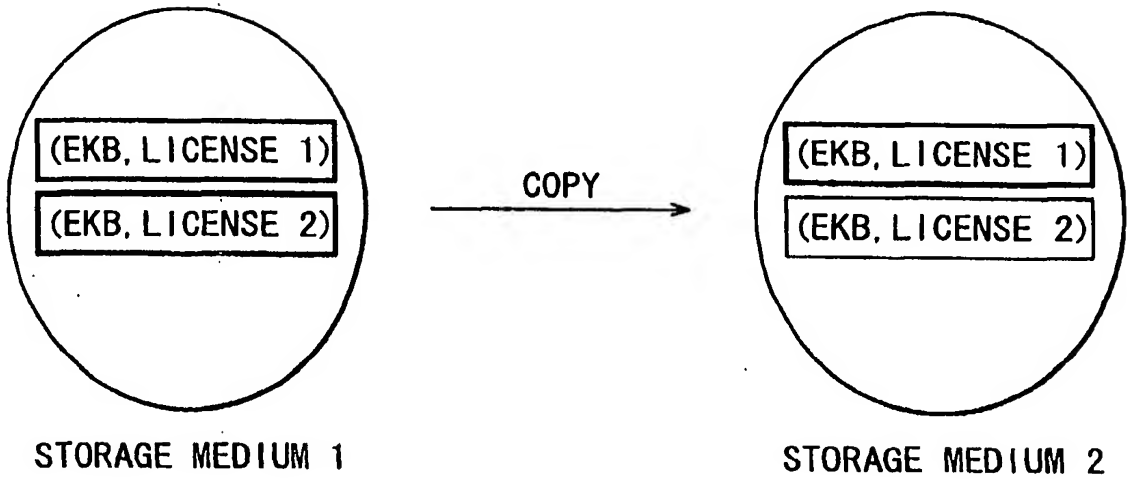
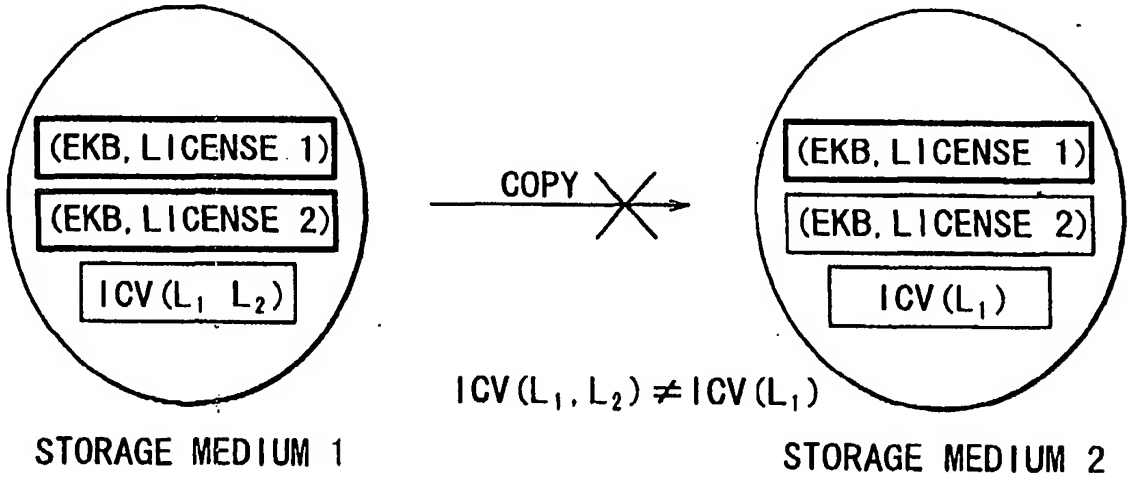


FIG. 50B



EP 1 307 000 A1

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP02/02957

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl[7] H04L9/08, G06F17/60

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int.Cl[7] H04L9/08, G06F17/60

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Toroku Jitsuyo Shinan Koho	1994-2002
Kokai Jitsuyo Shinan Koho	1971-2002	Jitsuyo Shinan Toroku Koho	1996-2002

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	JP 11-346210 A (Nippon Telegraph And Telephone Corp.), 14 December, 1999 (14.12.99), Par. Nos. [0001] to [0062] (Family: none)	1-6

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier document but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
04 July, 2002 (04.07.02)	23 July, 2002 (23.07.02)
Name and mailing address of the ISA/ Japanese Patent Office	Authorized officer
Facsimile No.	Telephone No.

Form PCT/ISA/210 (second sheet) (July 1998)